AN ASHTON-TATE SOFTWARE SUPPORT PUBLICATION NOVEMBER 1986

TechNotes

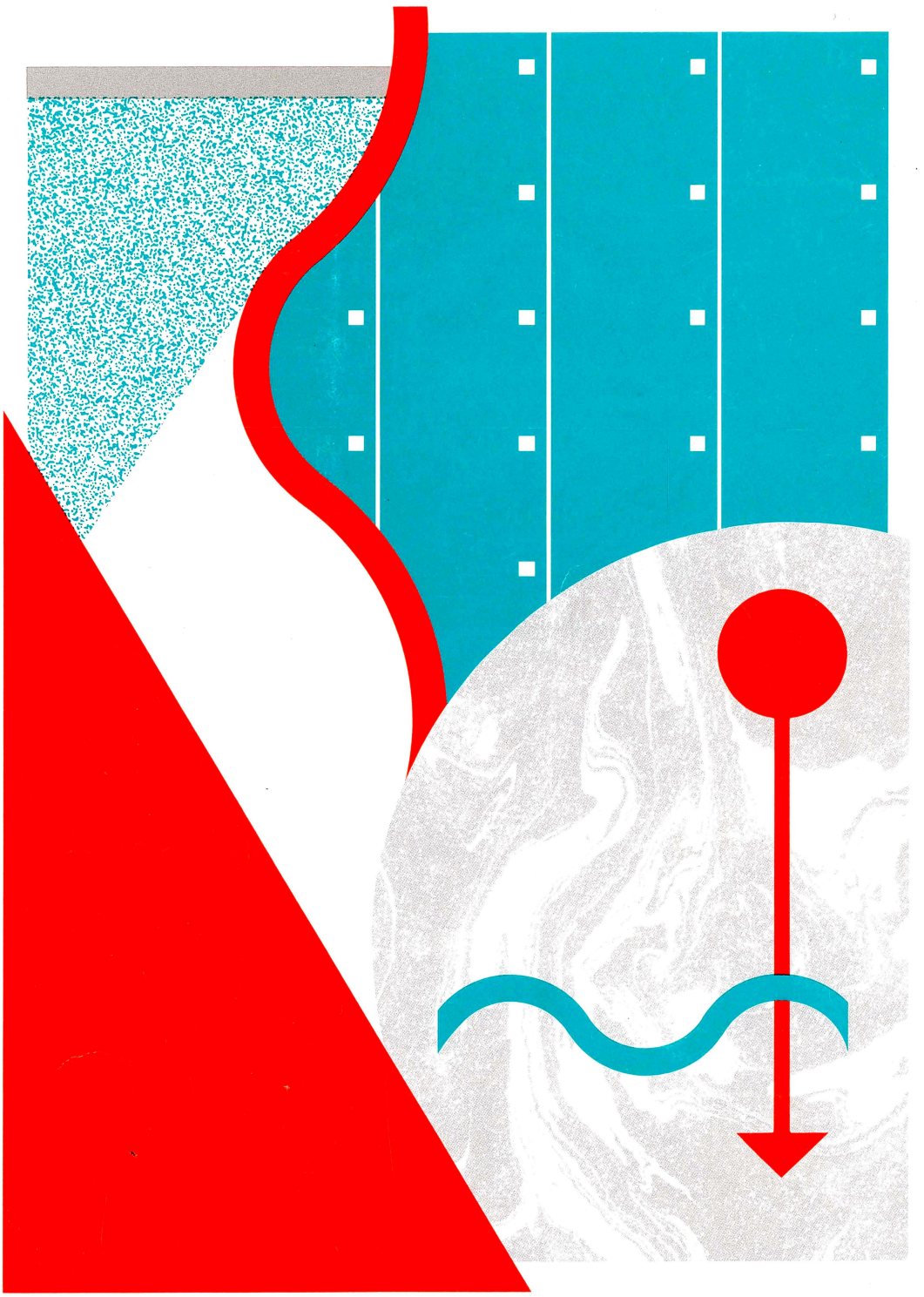
DEC 16 1986

Assorted Date Routines in dBASE III

dBASE III PLUS 1.1 Change Summary

Data Entry in Framework II

TCM File Report



ASHTON-TATE

TechNotes

TABLE OF CONTENTS

FEATURES

3 Assorted Date and Time Routines
by Software Support Staff and Brenda Johnson-Grau

15 Change Summary for dBASE III PLUS Version 1.0—Part I
by Gail Kulikowsky

18 Using MultiMate Advantage with dBASE III PLUS
By David Howlett and Karen Robinson

23 Sharing Data with Other Programs: DIF, SYLK, WKS
By Karen Robinson

27 TCM File Report
by Christopher White

31 A Simple Data-Entry Program for Spreadsheets and Databases
by Karen Robinson and Christopher White

DEPARTMENTS

1 Read Me First
by Laurie Miller

37 Usage Tips

48 Anomalies

49 On-Line Services

55 MultiMate Technical Support

61 Letters

63 Corrigenda

READ ME FIRST

by Laurie Miller

This month we are featuring date routines for dBASE III. We assembled several routines that operate with date type data, and presented them in the article (appropriately entitled) "Assorted Date Routines," written by Brenda Johnson-Grau. Brenda, one of our technical writers, is the newest addition to the ***TechNotes*** staff. Welcome aboard!

The routines themselves were written by our support technicians, and the ideas for many of them came from your phone calls to the Software Support Center. Included are routines that find a number of business days within a range of dates; return the time in a 12-hour format; return dates in the form of "Saturday, November 1, 1986;" average dates; index dates; calculate ages; and generate a scrolling date selector, to name a few. I'm sure you will find them helpful as you develop applications in dBASE III PLUS.

Also featured is the first part of a multi-part Change Summary for Version 1.1 (non-copy-protected) dBASE III PLUS, as promised last month. This Change Summary is important reading if you upgraded to 1.1, because it reports on all changes made to this version of the software from version 1.0.

Also in this issue are several informative articles that pertain to interfacing Ashton-Tate and MultiMate products with other software. Refer to "Using MultiMate Advantage with dBASE III PLUS" (by David Howlett and Karen Robinson) for information on setting up MultiMate Advantage as the text editor for dBASE III PLUS. "Sharing Data with Other Programs: DIF, SYLK, and WKS," also written by Karen, discusses how dBASE III PLUS data is represented when you COPY TO and APPEND FROM these formats. Finally, two articles in the **MultiMate Technical Support** section deal with converting Lotus 1-2-3 and DCA file formats to MultiMate.

For our Framework II readers, we have two articles of interest: "TCM File Report," by Christopher White, and "A Simple Data Entry Program for Spreadsheets and Databases," by Chris and Karen. The data entry article, the first of two parts, demonstrates data entry concepts by way of a simple FRED program.

The TCM article presents a program that utilizes **@sense** to extract all of the settings associated with a particular TCM driver from the Apps pull-down menu and generates a report of that information for ready reference.

TechNotes

Publisher
Ashton-Tate Software Support Center
Jay Hanson, Director
Roy Moore, Manager

Editor-in-Chief
Laurie Miller

Editors
Christopher White

Technical Writers
Brenda Johnson-Grau
Karen Robinson

Assistant Editor
Olivier Biggerstaff

Technical Staff
Larry Abare, Richard Augelli, Stanley Ballenger, Rodney Barbati, Jeffrey Bersin, Richard Bunter, Mike Carlisle, Ray Cortenbach, Fred Crownover, Michael Fiore, Rick Fillman, Charles Gehman, Ken Getz, Richard Goepel, Peter Gold, Mina Gussin, Doug Haman, Scott Haynes, Mike Hedblom, Clint Helton, Jack Hirsch, Errol Jackson, Matt Johnson,Rosaline Keenan, Gail Kulikowsky, Steven Kurasch, Jerry Levy, Chris Lo Vasco, Jeffrey McCrimon, Tom Mangelsdorf, Jerry Mendes, Vincent Mendillo, Elizabeth Miranda, Jeffrey Moore, Keith Mund, Joseph Perret, Jaspal Rattan, John Regen, Deborah Ridolfo, Daryl Saybold, Steve Silverwood, Joseph Stoltz, Naoma Stow, Ralph Wakefield, Lisa Wanek, Quinn Wildman, Roger Wegehoft,

Contributing Editors
Paul Bukauskas (International),
Liz Burke (MultiMate),
Steve Silverwood (On-Line Services)

Administrative Assistant
Kathy Smith

Illustrator
Stephen Goodman

Production Staff
D. A. Gray Bare, Tim Dunne, Jeff Elliott, Darrell Evers, Louis Hirsch, R. James Landers, Renée Maynard, Patty Pesek,

Product Development
Forefront Development Center:
Marilyn McMahon, Greg Stikeleather
Glendale Development Center:
Debby Moody

DISCLAIMER NOTICE

The information contained in this publication is designed to help you develop and improve your programming skills and to assist you in using Ashton-Tate products. Ashton-Tate assumes no responsibility whatsoever for the uses made of any of the software code in this issue, whether modified or unmodified.

If you discover what you believe is an anomaly (as defined under **Anomalies**) not presented in this issue or others, please send a detailed description of the possible anomaly to:

Ashton-Tate Software Support Center
20101 Hamilton Ave.
Torrance, CA 90502-1319

If the reported anomaly can be reproduced by our technicians, it may be added to a future issue of ***TechNotes***.

The policies that are described in ***TechNotes*** apply only to the United States (and possessions), Canada, and Mexico. Residents of other foreign countries should contact a local Ashton-Tate representative in their country for policy information and technical support.

Ashton-Tate and the authors will not be liable for special, incidental, consequential, indirect or other similar damages, even if we have been advised of the possibility of such damages. This means we are not responsible for damages or costs incurred as a result of loss of time, loss of data, loss of profits or revenues, or loss of use of the software. In addition, we are not responsible for damages or costs incurred in connection with obtaining substitute software, claims by others, inconvenience or similar costs.

No warranties, either expressed or implied, are made regarding the contents of this publication, its merchantability or fitness for any particular purpose, all of which are hereby expressly disclaimed. Note, however, that some states do not allow the exclusion or limitation of warranties or of direct, incidental or consequential damages, so these warranty limitations may not apply to you.

In no event will our liability for any damages to you or any other person ever exceed the price paid for your subscription to ***TechNotes***, regardless of any form of the claim.

Additional statements by agents, employees, distributors and dealers of Ashton-Tate do not constitute warranties by Ashton-Tate and do not bind Ashton-Tate.

TechNotes (ISSN 0883-6493) is an Ashton-Tate Software Support Center publication containing technical reference information on Ashton-Tate and MultiMate products.

Entire contents Copyright © Ashton-Tate 1986. All rights reserved. Commercial reproduction of the material appearing in ***TechNotes*** is forbidden without prior written permission. Direct all requests for reproduction permission to Ashton-Tate Software Support Center, 20101 Hamilton Ave., Torrance, CA 90502-1319 ATTN: Laurie Miller.

TechNotes is published monthly at a subscription price of $50.00 for one year. Additional rates: Canada and Mexico, $65.00; other countries, $75.00. Monthly issues are available in back issues for $10.00 each (1985 issues) and $7.00 each (1986 issues). ***TechNotes*** is published by the Ashton-Tate Software Support Center, 20101 Hamilton Ave., Torrance, CA 90502-1319. POSTMASTER: Send subscription inquiries and address changes to ***TechNotes*** Subscription Service, P.O. Box 3729, Escondido, CA 92025-0929.

dBASE II, dBASE III PLUS, and Framework II are registered trademarks of Ashton-Tate. dBASE, RunTime, FRED, and ASHTON-TATE are trademarks of Ashton-Tate. Some of the trademarks of other companies mentioned in this issue appear for identification purposes only.

The Ashton-Tate Software Support Center's hours are 6:00 a.m. to 4:30 p.m., PST, Monday through Friday, except on Thursday when we close at 4:00 p.m. To reach our technicians for technical support, call (213) 329-0086. Be sure to have your software serial number ready for the technician.

Assorted Date and Time Routines

by Software Support Staff and Brenda Johnson-Grau

This month we present a series of assorted date and time routines that illustrate ways to handle dates for business calendar needs. Some are new and a number of others were published in previous issues of ***TechNotes***. We have brought them together to give a more authoritative list of techniques and programs. We hope you find them useful.

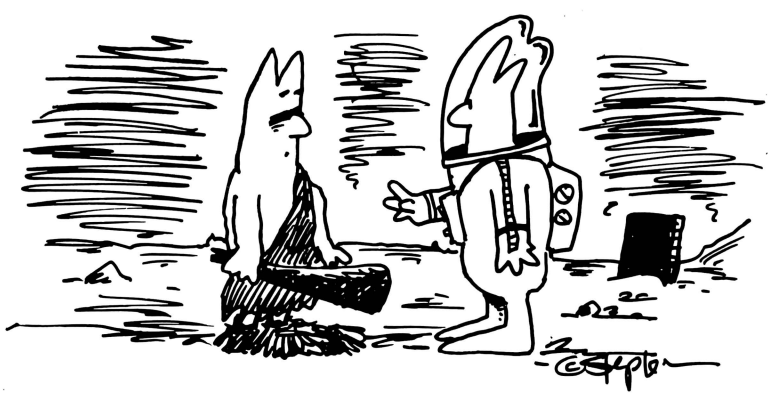
Setting the System DATE and TIME

There is no command in dBASE III or dBASE III PLUS to change the system time and date. To change either or both of them, RUN the DOS programs DATE and TIME. The desired date and time can be placed in a memory variable and passed to the operating system. For example,

```
* ---Set DOS system date.
today = "11-01-86"
RUN DATE &today

* ---Set DOS system time.
now = "08:00"
RUN TIME &now
```

Excerpted from the March, 1985 issue of ***TechNotes***.



Date Formatting

To create a custom date display, you can convert a date variable to a formatted string by using SUBSTR(), STR(), and date functions in combination. If, for example, you have a date and you want the output to look like "January 25, 1984," the expression that makes the conversion is,

```
CMONTH( <date> ) + " " + STR(DAY( <date> ), 2) +;
                   ", " + STR(YEAR( <date> ), 4)
```

In this expression, the first portion of the expression,

```
CMONTH( <date> )
```

returns the month name as a character string. The second portion of the expression,

```
STR(DAY( <date> ), 2)
```

returns the numeric day of the month as a character string. The last portion of the expression,

```
STR(YEAR( <date> ), 4)
```

returns the year as a four-character string. The results of this last expression are concatenated to the month and day. This expression can be used as the field contents in a REPORT FORM, a LABEL FORM, or any other display command that accepts a character expression such as ? or @...SAY.

General Date Formats

The following, Fdate.PRG, is a general purpose date formatting procedure. It takes two inputs, a date variable to format and a numeric code designating the format desired, and returns one output, a character variable containing the formatted date string. Call Fdate.PRG with the following general syntax.

```
* ---Receiving variable must exist prior to call.
 <return variable> = ""
DO Fdate WITH <date to format>,;
              <format code>,;
              <return variable>
```

Fdate.PRG supports several date formats. Pick from the following list using the position in the list as the format code parameter when you DO Fdate.

1. Mon DD, YYYY
2. Mon YYYY
3. Mon DD
4. Month DD, YYYY
5. DD-Mon-YYYY

Note that the day format for formats 1, 3, and 4, do not include a leading zero. If the day of the month is less than ten, the day will be a single digit number. For format 5, the day format includes a leading zero for a day of the month less than ten.

```
* Program..: Fdate.PRG
* Author...: Christopher White
* Date.....: November 1, 1986
*           published originally January 25, 1985
* Version..: dBASE III PLUS
* Note(s)..: This program takes a date variable passed to it
*           with the PARAMETERS phrase and returns the date
```

```
*         as a formatted character string.
*
PRIVATE ALL
PARAMETERS date, code, datestrg
DO CASE
CASE code = 1
     * ---"Nov 1, 1986"
     datestrg = SUBSTR(CMONTH(date), 1, 3) + " " +;
                LTRIM(STR(DAY(date), 2)) + ", " +;
                STR(YEAR(date), 4)
CASE code = 2
     * ---"Nov 1986"
     datestrg = SUBSTR(CMONTH(date), 1, 3) + " " +;
                STR(YEAR(date), 4)
CASE code = 3
     * ---"Nov 1"
     datestrg = SUBSTR(CMONTH(date), 1, 3) + " " +;
                LTRIM(STR(DAY(date), 2))
CASE code = 4
     * ---"November 1, 1986"
     datestrg = CMONTH(date) + " " +;
                LTRIM(STR(DAY(date), 2)) + ", " + STR(YEAR(date), 4)
CASE code = 5
     * ---"01-Nov-1986"
     datestrg = LEFT(DTOC(date), 2) + "-" +;
                SUBSTR(CMONTH(date), 1, 3) + "-" +;
                STR(YEAR(date), 4)
 OTHERWISE
     datestrg = "Error"
ENDCASE
RETURN
* EOP Fdate.PRG
```

Excerpted from the March, 1985 issue of ***TechNotes***.

Working Days

Weekday1

The following two programs, Weekday1.PRG and Weekday2.PRG, show how to account for weekend days when determining a number of working days. Weekday1.PRG allows you to select a start date and a number of days and will provide the number of working days **inclusive.**

To setup Weekday1.PRG or Weekday2.PRG, first CREATE Holiday.DBF with the following structure,

Structure for database: Holiday.DBF

Field	Field Name	Type	Width	Dec
1	Holiday	Date	8	

Then, APPEND all the holidays that span the range of dates in which you will be working. This means that if you are working with a span of two years, you must have APPENDed two records for each holiday, one for each specific date.

To execute Weekday1.PRG, use the following general syntax after initializing a return variable for the number of weekdays.

ret_val = 0
DO Weekday1 WITH <start date>, <number of days>, ret_val

```
* Program ...: Weekday1.PRG
* Author ....: Kenneth N. Getz
* Date ......: November 1, 1986
* Version ...: dBASE III PLUS
* Note(s) ...: Program to find number of working days given
*              a starting date and the numbers of days hence.
*
PRIVATE ALL
PARAMETERS start, days, weekdays
* ---Get starting date to be a week day.
DO WHILE MOD(DOW(start) - 1, 6) = 0
   start = start + 1
   days  = days - 1
ENDDO
* ---Subtract weekend days.
weekdays = days - INT(days / 7) * 2
* ---Find ending day.
endday = DOW(start + days - 1)
* ---Subtract for each weekend day.
IF MOD(endday - 1, 6) = 0
   weekdays = IIF(endday = 1, weekdays - 2, weekdays - 1)
ENDIF
USE Holiday
* ---Data file storing holidays.
COUNT TO counter FOR holiday >= start;
                 .AND. holiday <= start + days;
                 .AND. MOD(DOW(holiday) - 1, 6) <> 0
USE
* ---Subtract number of holidays.
weekdays = weekdays - counter
RETURN
* EOP Weekday1.PRG
```

Weekday2

Weekday2.PRG is a revision of Weekday1.PRG that allows you to specify a range of dates and returns the number of work days between them. To execute Weekday2.PRG, use the following general syntax after initializing a return variable for the number of work days returned from the routine.

ret_val = 0
DO Weekday2 WITH <start date>, <end date>, ret_val

```
* Program ...: Weekday2.PRG
* Author ....: Kenneth N. Getz
* Date ......: November 1, 1986
* Version ...: dBASE III PLUS
* Note(s) ...: Program to find number of working days given
*              a starting and ending date. Includes both end
*              dates.
*
PRIVATE ALL
PARAMETERS start, enddate, weekdays
days = enddate - start + 1
```

```
* ---Get starting date to be a week day.
DO WHILE MOD(DOW(start) - 1, 6) = 0
   start = start + 1
   days  = days  - 1
ENDDO
* ---Subtract off weekend days.
weekdays = days - INT(days / 7) * 2
* ---Find ending day.
endday = DOW(start + days - 1)
IF MOD(endday - 1, 6) = 0
   * ---Subtract for each weekend day.
   weekdays = IIF(endday = 1, weekdays - 2, weekdays - 1)
ENDIF
USE Holiday
Count TO counter FOR holiday >= start;
                 .AND. holiday <= start + days;
                 .AND. MOD(DOW(holiday)-1,6) <> 0
USE
weekdays = weekdays - counter
RETURN
* EOP Weekday2.PRG
```

Twelve-Hour Time Format

If you find the standard twenty-four—hour clock confusing, you may want to use the following procedure, Ampm.PRG, to display the time in the more familiar twelve-hour format.

Ampm.PRG works by using the SUBSTR() function to separate the hours and minutes from the time string you pass. These elements are then converted to the twelve-hour format. To execute Ampm.PRG, use the following general syntax after initializing a return variable,

ret_val = ""
DO Ampm WITH <time string>, ret_val

```
* Program ...: Ampm.PRG
* Author ....: Ray Cortenbach and Kenneth N. Getz
* Date ......: November 1, 1986
* Note(s)....: Returns time in 12-hour format (midnight is am,
*              noon is pm).
*
PRIVATE ALL
PARAMETERS m_time, hours
hours   = SUBSTR(m_time, 1, 2)
minutes = SUBSTR(m_time, 3, 3)
ampm    = IIF(hours >= "12"," pm"," am")
clock_hours = MOD(VAL(hours),12)
hours   = IIF(clock_hours = 0, "12", LTRIM(STR(clock_hours)))
hours   = hours + minutes + ampm
RETURN
* EOP Ampm.PRG
```

If you need this time format for a REPORT or LABEL FORM, you can use the following expression instead of Ampm.PRG.

```
IIF(VAL( <time> ) < 12, <time> + " am",;
   IIF(VAL( <time> ) = 12, <time> + " pm",;
      STR(VAL( <time> ) - 12 ,2) + SUBSTR( <time> , 3) + " pm"))
```

Displaying Days

Another date format you may need is the form "Friday November 1st, 1986." The following program converts a date to a character string and adds the appropriate "th", "st", "nd", "rd" to the day. To execute Day.PRG, use the following general syntax after initializing a return variable.

```
ret_val = ""
DO Day WITH <date>, ret_val
```

```
* Program ...: Day.PRG
* Author ....: Ray Cortenbach and Kenneth N. Getz
* Date ......: November 1, 1986
* Note(s)....: Returns date in format "Friday November 1st, 1986"
*
PRIVATE ALL
PARAMETERS m_date, current
daynum = DAY(m_date)
m_day  = LTRIM(STR(daynum, 2))
DO CASE
CASE daynum = 2 .OR. daynum = 22
     m_day = m_day + "nd"
CASE daynum = 3 .OR. daynum = 23
     m_day = m_day + "rd"
CASE MOD(daynum, 10) = 1 .AND. daynum <> 11
     m_day = m_day + "st"
OTHERWISE
     m_day = m_day + "th"
ENDCASE
current = CDOW(m_date) + " " + CMONTH(m_date) + " " +;
          m_day + ", " + STR(YEAR(m_date), 4)
RETURN
* EOP Day.PRG
```

Averaging Dates

Sometimes you may need to AVERAGE date fields in a database file. For example, you may want to know the average date of employment for all employees or the average date of maturity for a group of bonds.

Since the date data type is handled differently in dBASE III PLUS than the numeric data type, the AVERAGE command does not support averaging of dates but does numerics. To perform the AVERAGEing operation, you must convert the date fields you want to AVERAGE to a numeric values. To do this, subtract one of the earliest dates in the dBASE III PLUS calendar, 01/01/0001 from your date field and then the result can then be AVERAGEd. After AVERAGEing of the numeric values is complete, convert the result back to a date by adding it to the original date. The following sequence of commands demonstrates how to do this:

```
USE <database file>
AVERAGE <your date field> - CTOD("01/01/0001") TO avg
avgdate = CTOD("01/01/0001") + avg
```

You can also use this procedure that returns the average date from a date field that you pass as a parameter in place of these commands. To execute it, use the following general syntax after initializing a return variable.

```
ret_val = CTOD("01/01/01")
DO Dateavg WITH "<date field to average>", ret_val
```

```
* Program ...: Dateavg.PRG
* Author ....: Brenda Johnson-Grau
* Date ......: November 1, 1986
* Note(s) ...: Procedure to return average of date field in a
*              database file.
*
PRIVATE ALL
PARAMETERS date_fld, avgdate
AVERAGE &date_fld. - CTOD("01/01/0001") to avg
avgdate = CTOD("01/01/0001") + avg
RETURN
* EOP Dateavg.PRG
```

Calculating Ages

If you need to calculate and display a person's age according to the current date (rather than estimate from the year alone), you need to test both current month for the birth month and the current date for the birth date. The following procedure, Age.PRG, takes a date passed to it and displays the person's age in the format: "x years old", where x is the calculated age.

```
* Program ...: Age.PRG
* Author ....: Beth Miranda
* Date ......: November 1, 1986
* Version....: dBASE III
* Note(s) ...: Calculates a person's age from today's date.
*
PARAMETERS birthdate
age = 0
currtmon = MONTH(DATE())
birthmon = MONTH(birthdate)
IF YEAR(DATE()) > YEAR(birthdate)
   age = YEAR(DATE()) - YEAR(birthdate)
   IF currtmon < birthmon .OR. ( currtmon = birthmon;
                .AND. DAY(DATE()) < DAY(birthdate))
      age = age - 1
   ENDIF
ENDIF
? STR(age,4) + " years old"
RETURN
* EOP Age.PRG
```

In dBASE III PLUS, you can replace the program above with the following expression for use in REPORT and LABEL FORMs.

```
? IIF(YEAR(DATE()) > YEAR(d2),;
    IIF(MONTH(DATE()) < MONTH(d2).OR.(MONTH(DATE()) = MONTH(d2);
        .AND.DAY(DATE()) < DAY(d2)),;
        YEAR(DATE())-YEAR(d2)-1,;
        YEAR(DATE())-YEAR(d2);
      ),;
      0;
    )
```

Be aware that this expression is 217 bytes long. 12 bytes are allocated for the "d2" variable name. You can expand the variable name length to 10 characters but the expression will be 253 characters, precariously close to the 254-character limit on the length of command lines. If you plan to use this expression keep your date variable lengths short.

Indexing Dates

INDEXing ON a date is quite easy. For example, the command:

INDEX ON < date field > TO < date index >

will produce an index in chronological order by year, month, and day.

However, a problem arises when INDEXing ON a compound key that includes a date field which is common in the majority of applications. A date is rarely the unique identifier or the sole determinant of order. Suppose that you have a transaction-oriented application. Typically, the transactions database file will be INDEXed ON the date of transaction and some form of customer identification. The logical order is time-oriented, but the customer order must be maintained. Or, the customer identification is the primary key and the date of transaction the secondary key—a type of ordering useful for invoicing, account history reporting, and aging analysis.

Multiple Key Ascending Order

Defining an INDEX expression as a compound key requires that the key expression evaluate to a single data type. In most instances, this data type will be character. The character data type offers concatenation, a powerful tool to combine elements of an expression, which preserves a left-to-right precedence of order.

When defining a character expression that includes a date variable, your natural inclination might be to use DTOC() to convert the date field to a character string, and in most cases it would be a valid assumption. INDEXing with the DTOC() function, however, does not yield a valid date order. Instead, DTOC() returns as a character value, the date in the format specified by the SET DATE command. The default is MM/DD/YY, the American format. You could SET DATE ANSI and then INDEX ON the DTOC() of the date field, but this approach has its hazards. In order to maintain the integrity of the index file, DATE must be SET ANSI whenever the index is updated. However, the display format preferred in the United States is not ANSI. A considerable amount of work would be required to make sure that every change to the database file is prefaced by a SET DATE ANSI and followed by a SET DATE AMERICAN for display operations.

The preferred method of INDEXing ON the date field with a compound key expression is to use the YEAR(), MONTH(), DAY(), and the STR() functions. This will create the correct hierarchy of order, converting the date to a character string. A example might look like,

```
* ---Create index file.
INDEX ON STR(YEAR(Date),4) + STR(MONTH(Date),2);
         + STR(DAY(Date),2) + Custnum TO DateNdx
* ---Look up entry in the index file.
transdate = CTOD(" / / ")
customer = SPACE(5)
@ 10,10 SAY "Enter transactions date " GET transdate
@ 11,10 SAY " Enter customer number " GET customer
READ
SEEK STR(YEAR(transdate),4) + STR(MONTH(transdate),2);
       + STR(DAY(transdate),2) + customer
```

Be sure that whenever you use STR() in an index expression, to specify a length parameter. Failure to do so results in the error message "Record not in index" whenever the index file is used.

Descending Order

INDEXing dates in descending order provides a new twist. dBASE III and dBASE III PLUS does not directly support a descending order index file as dBASE II does. Therefore, an algorithm must be devised to create a descending order and integrated with the date so that the hierarchy of order — year, month, and day — is preserved.

In creating a descending order index file, the general idea is to subtract the key value from a number larger than the largest possible key value and pass that to the index mechanism as the new key value. This reverses the order of keys, creating a descending order. To use this algorithm, a date must be converted into a number that is in the form YYYYMMDD. This can be done as follows:

```
YEAR( < date > ) * 10000 + MONTH( < date > ) * 100 + ;
DAY( < date > )
```

If today's date is 11/01/86, the result of the expression will be 19861101. Notice that the integrity of the date is preserved and the result is in a form that can be subtracted from a very large number.

To INDEX a date in descending order, assemble the final algorithm and execute it in an INDEX command. For example:

```
INDEX ON STR(9999999 - (YEAR( < date > )*10000;
 + MONTH( < date > )*100 + DAY( < date > )),8) TO < your index >
```

Essentially, INDEXing ON dates successfully requires some simple conversions and calculations.

Excerpted from the August, 1985 issue of ***TechNotes***.

Scrolling Date Selectors

Perhaps you would like to set up an application where you can easily select specific dates or periods. The following programs, Scroll1.PRG and Scroll2.PRG, allow you to do this. Each displays the date or date range in a reverse video window allowing you scroll forward or back in time with the **Uparrow** and **Dnarrow** keys. Pressing **Return** selects the date or date period displayed. Pressing **Esc** returns a blank or the seed date passed.

For both programs, you pass the row and column coordinates of the scroll window and a return variable as parameters.

Scroll Time Periods

Scroll1.PRG allows you to scroll through time periods beginning with the closest period to the system date. The period begins on the closest Monday to the current date and ends on the Friday of the next week. To change the length and the last day of the period, change the initial values of the memory variables "period" and "per_end".

Pressing **Esc** returns a blank date. Pressing **Return,** returns the beginning date of the time period your have selected.

To execute Scroll1.PRG, use the following commands,

```
ret_val = CTOD("  /  /  ")
DO Scroll1 WITH <row>, <col>, ret_val
```

```
* Program ...: Scroll1.PRG
* Author ....: Christopher White
* Date .......: November 1, 1986
* Note(s) ...: Selects a time period from a scrolling menu.
*
PRIVATE ALL
PARAMETERS row, col, seed
* ---Key definitions.
esc     = CHR(27)
uparrow = CHR(5)
dnarrow = CHR(24)
enter   = CHR(13)
* ---Initialize dates.
period  = 14
per_end = period - 3
start   = CTOD("01/08/1900")
seed    = DATE() - (MOD(DATE() - start, period) + period)
DO WHILE .T.
   SET COLOR TO N/W
   @ row, col SAY LEFT(CDOW(seed),3) + " " + DTOC(seed) +;
                  " thru " + LEFT(CDOW(seed + per_end), 3) +;
                  " " + DTOC(seed + per_end)
SET COLOR TO W/N
key = CHR(0)
* ---Get key.
DO WHILE .NOT. key $ esc + enter + dnarrow + uparrow
   key = UPPER(CHR(INKEY()))
ENDDO
DO CASE
CASE key = dnarrow
     seed = seed + period
CASE key = uparrow
     seed = seed - period
CASE key = enter
     EXIT
CASE key = esc
     seed = CTOD(" / / ")
     EXIT
   ENDCASE
ENDDO
RETURN
* EOP Scroll1.PRG
```

Scroll Days

Scroll2.PRG allows you to scroll through consecutive days beginning with the date you pass in the memory variable, "ret_val." Pressing **Return** returns the date you selected in the variable "ret_val." Pressing **Esc**, returns the seed date. Subsequent runs of the program uses the last value stored in "ret_val" for the seed date. This is useful so that you can carry forward selected dates.

To execute Scroll2.PRG, use the following commands.

```
ret_val = DATE()
DO Scroll2 WITH <row>, <col>, ret_val
```

```
* Program ...: Scroll2.PRG
* Author ....: Christopher White
* Date ......: November 1, 1986
* Note(s) ...: Selects a day from a scrolling menu.
*
PRIVATE ALL
PARAMETERS row, col, seed
* ---Key definitions.
esc     = CHR(27)
uparrow = CHR(5)
dnarrow = CHR(24)
enter   = CHR(13)
* ---Initialize dates.
start  = CTOD("01/08/1900")
b_date = seed
DO WHILE .T.
   SET COLOR TO N/W
   @ row, col SAY LEFT(CDOW(seed),3)+" "+DTOC(seed)
   SET COLOR TO W/N
   key = CHR(0)
   * ---Get key.
   DO WHILE .NOT. key $ esc + enter + dnarrow + uparrow
      key = UPPER(CHR(INKEY()))
   ENDDO
   DO CASE
   CASE key = dnarrow
        seed = seed + 1
   CASE key = uparrow
        seed = seed - 1
   CASE key = enter
        EXIT
   CASE key = esc
        seed = b_date
        EXIT
   ENDCASE
ENDDO
RETURN
* EOP Scroll2.PRG
```

Changing a Character Field to a Date Field

You can convert character fields that contain dates in the form "mm/dd/yy" or any of the formats specified by the SET DATE command to dBASE III PLUS date fields without losing the date value. You do this by MODIFYing the STRUCTURE of the database file and changing the field that contains the date information from character to date type. Saving this change converts all dates stored in the character field as "mm/dd/yy" directly to dBASE III PLUS date values.

This is useful when converting dBASE II database files to dBASE III PLUS.

Excerpted from the February, 1985, issue of ***TechNotes***.

Contributors

Steve Silverwood
Beth Miranda
Christopher White
Kenneth Getz
Ray Cortenbach
Brenda Johnson-Grau

Change Summary for dBASE III PLUS Version 1.0—Part I

by Gail Kulikowsky

This begins a multi-part series listing anomalies reported for dBASE III PLUS version 1.0 and fixed in dBASE III PLUS version 1.1. They are listed here so that dBASE III PLUS version 1.0 users will have a clear idea of which anomalies have been corrected and which are still outstanding.

Please note that this month's list does not include any of the anomalies fixed in the dBASE Administrator.

@...CLEAR TO

@...CLEAR TO, in dBASE III PLUS version 1.0, did not erase a single line from the screen, although the @...TO could draw a single line. The minimum vertical area @...CLEAR TO would erase was two lines.

In dBASE III PLUS version 1.1, you can use @...CLEAR TO to erase a single line on the screen.

Reported in the September 1986 issue of ***TechNotes***, page 42.

ASSIST

In dBASE III PLUS version 1.0, performing any full-screen edit on a record found with the Locate option of the Position menu disabled the Continue option. When you returned from a full-screen operation, Continue was no longer highlighted and so cannot be accessed.

In dBASE III PLUS version 1.1, you Continue from the Position menu after a full-screen edit operation.

Reported in the September 1986 issue of ***TechNotes***, page 43.

CREATE/MODIFY SCREEN

In dBASE III PLUS, if you were editing on the blackboard and you pressed **Ctrl-T** with the cursor on a line and column to the left of a special character (!,?,.,@), the computer would freeze, requiring a warm boot.

In dBASE III PLUS version 1.1, you can press **Ctrl-T** on the blackboard without problem.

Reported in the September 1986 issue of ***TechNotes***, page 44.

FIND

In dBASE III PLUS version 1.0, the FIND command with in-line comments (&&) would fail if there were spaces between the macro variable and the comments. For example, the command

```
FIND &memvar && Comment
```

would not FIND the value contained in the variable "memvar."

In dBASE III PLUS version 1.1, you can position in-line comments anywhere on the FIND command line without error.

Reported in the September 1986 issue of ***TechNotes***, page 44.

FOUND()

In dBASE III PLUS version 1.0, if you SET DELETED ON and performed FIND or SEEK on a key with multiple occurrences and where the first occurrence was deleted, FOUND() returned a .F., even though a matching record was found.

In dBASE III PLUS version 1.1, you can use the FOUND() function after SEEK or FIND without error when SET DELETED is ON.

Reported in the April issue of ***TechNotes***, page 48.

INDEX

1. In dBASE III PLUS version 1.0, INDEX or REINDEX sometimes returned the error message "File is not accessible" and damaged index files in use. This occurred when the length of the key expression and path to the .NDX file exceeded 100 characters. This did not occur with small database files, but adding records eventually produced the error. Note that not all key expression and path name combinations exceeding 100 characters produced this error.

   In dBASE III PLUS version 1.1, you can create an index file where the key expression and path length together exceed 100 characters.

   Reported in the September 1986 issue of ***TechNotes***, page 45.

2. In dBASE III PLUS version 1.0, SET CENTURY ON and DTOC() were not compatible for the purpose of creating an index file. INDEXing ON the DTOC() of a date field with SET CENTURY ON appeared to index the file correctly. The resulting index file could be accessed, the FIND or SEEK commands could be used, but any attempt to BROWSE, EDIT, or GOTO a record produced the error message, "Record not in index."

   In dBASE III PLUS version 1.1, you can create a valid index file with SET CENTURY ON and using the DTOC() of a date field as the key expression.

   Reported in the April 1986 issue of ***TechNotes***, page 48.

REINDEX

In dBASE III PLUS version 1.0, REINDEXing on large database files of 1500 or more records could produce the error message "File is not accessible." Trying to use the index after this produced the error message "INDEX damaged. REINDEX should be done before using data."

In dBASE III PLUS version 1.1, REINDEXing large database files produces a valid index without error.

Reported in the September 1986 issue of ***TechNotes***, page 45.

SET PRINT ON

In dBASE III PLUS version 1.0, there were reported instances of the "Printer not ready" error message appearing on parallel as well as serial printers. This usually happened when you used SET PRINT ON, SET DEVICE TO PRINT, TO PRINT, or SET PRINTER TO to output to the printer. Printing from DOS or other software packages worked properly.

This could also occur when you were printing multiple reports to a parallel port. The problem usually occurred at a page eject or when the computer sensed the printer was off-line for an extended period of time.

In dBASE III PLUS version 1.1, you can print to parallel printers without receiving erroneous error messages.

Reported in the September 1986 issue of ***TechNotes***, page 46.

TOTAL

In dBASE III PLUS version 1.0, TOTALing a database file with no numeric fields could produce the error message "Internal error: EVAL work area overflow" or "Internal error: Illegal opcode". The target database file was created and control returned to the dot prompt. On other occasions, this type of TOTAL returned the error message "Internal error: EVAL work area overflow", the DOS error message "Divide overflow," and then the computer would freeze, requiring a cold boot.

dBASE III PLUS version 1.1 now allows you to TOTAL database files with no numeric fields correctly.

Reported in the September 1986 issue of ***TechNotes***, page 47.

TRANSFORM()

In dBASE III PLUS version 1.0, using TRANSFORM() with an argument combining the @(and @B functions displayed extraneous characters following the output. For example,

```
? TRANSFORM(-10,'@(B9999')
(10.00 )) <---Result.
```

The extra characters might have been another parenthesis, a field name, or the argument from a previous function.

In dBASE III PLUS version 1.1, you can combine the @(and @B functions within the TRANSFORM() function and get expected results.

Reported in the September 1986 issue of ***TechNotes***, page 47.

Using MultiMate Advantage with dBASE III PLUS

by David Howlett and Karen Robinson

Many users prefer to use a text editor other than the dBASE III PLUS word processor for editing .PRG files and memo fields. The article "Using Other Programs with dBASE III," in the April 1985 issue of ***TechNotes***, discussed how to set up dBASE III to use external word processors that can create and edit ASCII text files. If your word processor cannot create or edit ASCII text files or does not support subdirectories, you need a alternative method to transfer information to and from your word processor. MultiMate Advantage, for example, must reside in the current directory when you use it. You cannot execute it by setting the DOS PATH command or by using a path statement. In addition, MultiMate Advantage cannot create or edit ASCII text files directly, although it does have a conversion utility (UTIL) that converts ASCII text files to the MultiMate Advantage format and vice versa.

In this article, we'll show you how to use MultiMate Advantage with dBASE III PLUS by setting up a DOS batch file. Instead of making a direct call to the word processor, dBASE III PLUS can call a batch file that carries out the processes necessary to run MultiMate Advantage in another subdirectory and to execute the file conversion utility.

How dBASE III PLUS Communicates with External Word Processors

dBASE III PLUS accesses an external text editor or word processor by sending commands to DOS. These commands are specified by keywords in CONFIG.DB and they differ depending on whether you are editing a memo field or MODIFY COMMAND. When you execute MODIFY COMMAND, dBASE III PLUS sends to DOS whatever text follows TEDIT = in CONFIG.DB plus the name of your .PRG file. If CONFIG.DB contains the command

```
TEDIT = WS
```

and from the dot prompt you type

```
MODIFY COMMAND Test
```

then dBASE III PLUS sends this command line to DOS:

```
WS Test.PRG
```

For memo fields, dBASE III PLUS sends to DOS whatever text follows WP = in CONFIG.DB. dBASE III PLUS doesn't, however, allow you to edit the .DBT file directly. When you select a memo field for editing, dBASE III PLUS creates a temporary file called DBASEDIT.TMP and allows you to edit this file. If the memo field has data in it, dBASE III PLUS creates DBASEDIT.TMP, copies the memo information into it, and then passes control to the specified word processor with DBASEDIT.TMP as the filename. If you edit a memo field that is empty, dBASE III PLUS simply passes the DBASEDIT.TMP filename to your word processor. When you finish editing, the new version of the memo field is written back to the .DBT file.

Using an External Word Processor

These are the general requirements for using an outside word processor in place of the dBASE III PLUS text editor:

1. The dBASE III PLUS file CONFIG.DB must contain one or both of the keywords, TEDIT and WP. TEDIT selects the text editor used for MODIFY COMMAND and WP selects the word processor used for memo fields. The command syntax is as follows:

   TEDIT = <word processor filename>
   WP = <word processor filename>

   If you are using DOS 3.x and if your word processor can handle path names, you can precede the filename in CONFIG.DB with a path name. DOS 2.x does not support this. In either version of DOS, file extensions are ignored because the DOS order of precedence is always followed: .COM files first, then .EXE, then .BAT.

2. If your word processor cannot create or edit ASCII text files, it must have a conversion process that converts an ASCII text file to its own file format and back again.

3. Your computer needs to have enough RAM to run your word processor within dBASE III PLUS. This memory requirement varies, depending on the word processor and the version of DOS.

Setting Up a Batch File

The contents of your batch file will depend on the capabilities of your word processor. The steps and examples below outline the general structure of a typical batch file.

1. If your word processor does not support subdirectories and is not located in the dBASE III PLUS directory, include a command in the batch file to copy the text file to the word processor's directory. For example,

   COPY %1 <drive> <directory>

   %1 is a DOS argument variable that accepts the name of the dBASE III PLUS text file that is passed to it. The name of the text file is passed from dBASE III to the batch file and is stored in %1.

2. Once the file is copied, log onto the word processor's directory:

   CD\<directory>

3. If your word processor does not edit ASCII text files, convert the text file to the correct file format. To convert an ASCII text file to the MultiMate format, the command is:

   UTIL F C %1 ASCII < filename > MM

4. Next, load the word processor and edit the text file:

   WP < filename >

   Refer to the MultiMate Advantage *Advanced User's Guide* for further information on using UTIL.

5. After exiting the word processor, convert the file back to an ASCII text file format, if necessary. For MultiMate Advantage, the command line is as follows:

   UTIL F C < filename > MM %1 ASCII

6. Copy the text file back to the dBASE III PLUS directory:

   COPY %1 < drive > < directory >

7. Return to the dBASE III PLUS directory:

   CD\ < directory >

8. Exit the batch file and return to dBASE III PLUS:

   EXIT

For further information on these DOS commands, refer to your DOS manual.

Using MultiMate Advantage as the dBASE III PLUS Editor

Medit.BAT, the DOS batch file that is listed below, specifically allows you to use MultiMate Advantage as the dBASE III PLUS word processor. The program makes three assumptions:

1. MultiMate Advantage is not in the dBASE III PLUS directory.
2. The conversion program, UTIL, is installed correctly (see "Setup" below) and is in the same directory as MultiMate Advantage.
3. Your computer has enough RAM (320K) to run MultiMate Advantage within dBASE III PLUS.

When UTIL converts the dBASE III PLUS text file to MultiMate format, it strips the hard carriage returns within paragraphs, placing a MultiMate "format line" symbol after every 256 characters. Then when UTIL converts the MultiMate document back to an ASCII text file, it places a carriage return at the end of each line. Carriage returns at the end of each line can cause formatting problems when printing from dBASE III PLUS. In order to avoid these problems, your MEMOWIDTH setting in dBASE III PLUS should be equal to or greater than the right margin you set in MultiMate Advantage. The default memo width in dBASE III PLUS is 50 and the default right margin in MultiMate Advantage is 65. For consistency, you may want to SET the MEMOWIDTH to 65 in your CONFIG.DB file with the statement MEMOWIDTH = 65.

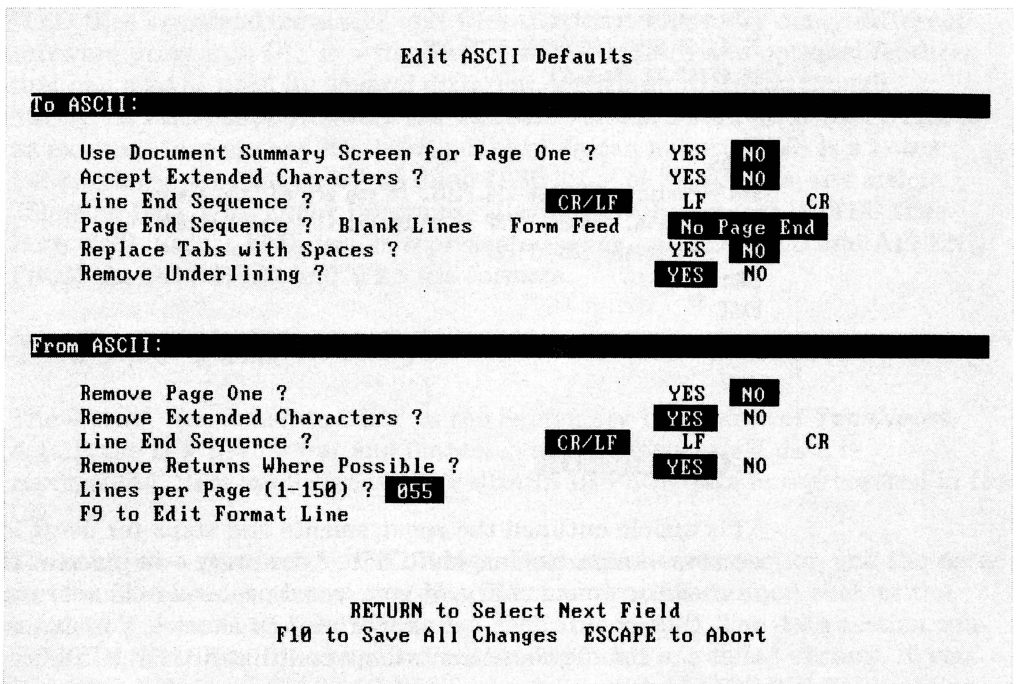
Remember also, if you set the width of a REPORT FORM column that contains a memo field to less than the right margin in MultiMate Advantage, you will get incorrect wrapping because of these embedded carriage returns. Therefore, the MEMOWIDTH and REPORT FORM column width for a memo field must be less than or equal to the right margin set in MultiMate Advantage.

Setup

To set up, create Medit.BAT with your dBASE word processor or with MultiMate Advantage. Remember to convert the MultiMate file to an ASCII text file with UTIL. Copy the file to the dBASE III PLUS directory. Then add file one or both of the following command lines to your CONFIG.DB file:

```
TEDIT = MEDIT
WP = MEDIT
```

In your MultiMate Advantage directory, install UTIL with these parameters:



Once MultiMate loads, press **F10** twice to move beyond the "Document Summary Screen" and the "Modify Document Defaults" screen.

MEDIT.BAT

The drive letters and directory names, printed in **bold** type, should be changed to the correct names for your system. REMark statements are used to clarify the process; you don't have to include them in your file.

```
REM ---Medit.BAT
ECHO OFF
CLS
ECHO Transferring control to MultiMate...
REM ---Copy the text file, whether it's DBASEDIT.TMP or your
REM ---program file, to the MultiMate subdirectory.
IF EXIST %1 COPY %1 C:\Multi\%1>NUL
REM ---Log onto MultiMate subdirectory.
C:
CD\Multi
```

```
REM ---Use UTIL to convert the file to a MultiMate
REM ---document file and rename it to MEMO.DOC.
IF EXIST %1 UTIL F C %1 ASCII MEMO.DOC MM
IF ERRORLEVEL 1 ECHO An error was found.
GOTO END
REM ---Execute MultiMate and edit the file.
WP MEMO.DOC
REM ---Convert the MultiMate file back to an ASCII text
REM ---file and rename it back to its original filename.
IF EXIST MEMO.DOC UTIL F C MEMO.DOC MM %1 ASCII
IF ERRORLEVEL 1 ECHO An error was found.
GOTO END
CLS
ECHO Transferring control to dBASE III PLUS...
REM ---Copy the text file back to the dBASE III PLUS
REM --- subdirectory.
IF EXIST %1 COPY %1 C:\D3>NUL
REM ---Erase MEMO.DOC and the text file from MultiMate
REM ---subdirectory.
IF EXIST MEMO.DOC ERASE MEMO.DOC
IF EXIST %1 ERASE %1
REM ---Change back to dBASE III PLUS drive and directory.
C:
CD\D3
REM ---Return to dBASE III PLUS. If you are editing a memo
REM ---field, DBASEDIT.TMP is automatically incorporated
REM ---into the .DBT file.
END:
EXIT
```

Conclusion

This article outlined the requirements and steps for using external word processors—in particular, MultiMate Advantage—in place of the dBASE III PLUS text editor. In summary, if your word processor cannot create or edit ASCII text files or if it does not support subdirectories, you can create a batch file that can handle these limitations and that dBASE III PLUS can call from CONFIG.DB.

Sharing Data with Other Programs: DIF, SYLK, WKS

by Karen Robinson

dBASE III PLUS can COPY TO and APPEND FROM several file types: SDF, DELIMITED, DIF, SYLK, and WKS. SDF (Standard Data Format) and DELIMITED files are standard ASCII text files that can be used by many different software programs. DIF is a file format with standard and optional features that can also be used by several different software programs, though dBASE III PLUS supports only the VisiCalc version 1 format of DIF. SYLK is an external format that Microsoft's Multiplan can import. WKS is a Lotus 1-2-3, release 1A format. In the June 1985 issue of ***TechNotes***, the article "Sharing Data with Other Programs" discussed SDF and DELIMITED files. Here we'll discuss how data is represented when you COPY TO and APPEND FROM the DIF, SYLK, and WKS file formats.

The DIF File

The article "Exporting to DIF," in the September 1986 issue of ***TechNotes***, details the DIF file format and discusses how Framework II data is represented. Here we'll discuss how dBASE III PLUS data is represented in the DIF file.

There are two parts to the DIF file structure: the header section and the data section. The header section contains file structure information such as the number of records and the number of fields per record. The data section contains the data. Records are called *tuples* and fields are called *vectors*. If you COPY TO DIF from dBASE III PLUS, the first record of the DIF file contains the field names from the dBASE III PLUS database file structure.

In the DIF file format, all data types are stored as either character or numeric. The table below outlines how dBASE III PLUS data types are represented when copied to a DIF file:

dBASE III Data Type	**DIF Data Type**
Character	Character
Date	Numeric, in the format YYYYMMDD
Numeric	Numeric
Logical	Numeric: 1 = true, 0 = false
Memo	Does not export memo fields

This dBASE III PLUS database file:

Record#	NAME	LOGICAL	DATE	NUMBER
1	Michael	.T.	07/21/86	123.45
2	Karen	.F.	10/01/86	129.88

was used to create this DIF file structure:

```
0,1
""
TUPLES      Number of .DBF
0,3         records
""
VECTORS     Number of .DBF
0,4         fields
""
DATA
0,0
-1,0
            Beginning of
BOT         record marker
1,0
"NAME"
1,0         First record contains
"LOGICAL"   .DBF field names
1,0
"DATE"
1,0
"NUMBER"
-1,0        Beginning of
BOT         record marker
1,0
"Michael"
0,1
TRUE        Data for Record 1
0,19860721
V
0,123.45
V
-1,0        Beginning of
BOT         record marker
1,0
"Karen"
0,0
FALSE       Data for Record 2
0,19861001
V
0,129.88
V
-1,0        End-of-data
EOD         marker
```

DIF Header Section

Record 1 (tuple)

Record 2 (tuple)

Record 3 (tuple)

DIF Data Section

Remember that the first record (tuple) in the DIF file contains the field names from the database file. If you have COPYed TO a DIF file and then at some later time APPEND FROM it, delete this tuple; otherwise, the first record APPENDed will contain the dBASE III PLUS database file header information.

The SYLK File

SYLK is a Multiplan file type, called a *symbolic* file. Like the DIF file, it is an ASCII text file with a header section and a data section. The header section includes file structure information, such as the number of records and fields, and the data section contains the data. The first record in the data section has the field names from the dBASE III PLUS database file. SYLK files store two data types: numeric and character. The table below outlines how dBASE III PLUS data types are represented:

dBASE III Data Type	**SYLK Data Type**
Character	Character
Date	Character, in the format YYYYMMDD
Numeric	Numeric
Logical	Character: T = true, F = false
Memo	Does not export memo fields

If we use the same sample database as in the DIF example, we find that the SYLK file looks like this:

```
ID;PDB3
F;DG0G10                  Y = number of records   ] Header
B;Y3;X4                 → X = number of fields    ]
C;Y1;X1;K;"NAME"          ]                       ]
C;X2;K"LOGICAL"           ] Record 1 contains     ]
C;X3;K;"DATE"             ] field names           ]
C;X4;K;"NUMBER"           ]                       ]
C;Y2;X1;K;"Michael"       ]                       ] Data
C;X2;K"T"                 ] Record 2              ]
C;X3;K"19860721"          ]                       ]
X;X4;K123.45              ]                       ]
C;Y3;X1;K"Karen "         ]                       ]
C;X2;K"F"                 ] Record 3              ]
C;X3;K"19861001"          ]                       ]
C;X4;K129.88              ]                       ]
W;N1;A1 1
E                       → End-of-file marker
```

APPENDing FROM a SYLK File

Successfully APPENDing FROM a SYLK file depends on how the data is entered into the Multiplan spreadsheet. Before creating the SYLK file from your spreadsheet, review the following list of limitations:

1. Data cannot be the result of a formula, such as SUM or AVERAGE. In addition, data must not be entered using cursor pointing or absolute cell referencing.
2. Any cell that is a result of an error, such as a division by zero, will result in the dBASE III PLUS error message "Invalid SYLK File Format" when APPENDing FROM.
3. If any of the cells in the first column of the Multiplan spreadsheet are empty, dBASE III PLUS will not correctly APPEND the data.

The WKS File

The WKS format is a 1-2-3, release 1A file. It is stored as a binary file and so the structure is not available in a form that meaningful. We can, however, still discuss how data is represented when you COPY TO and APPEND FROM the WKS format. This table shows how dBASE III PLUS data is represented in a WKS file:

dBASE III Data Type	WKS Data Type
Character	Character
Date	Character, in the format YYYYMMDD
Numeric	Numeric
Logical	Character: T = true, F = false
Memo	Does not export memo fields

APPENDing FROM a WKS file

When APPENDing FROM a WKS file, consider the following situations:

1. dBASE III PLUS logical fields will not accurately represent the data from the WKS file. Any data type APPENDed into a dBASE III PLUS logical field becomes false (.F.), regardless of whether the data is a 1-2-3 logical data type and what value it contains.
2. 1-2-3 date data types do not APPEND correctly into dBASE III PLUS date fields. To correctly APPEND dates, 1-2-3 date data types must be either character or numeric and in the format YYYYMMDD.
3. Values that are entered directly into the 1-2-3 worksheet APPEND correctly into dBASE III PLUS. However, values that are a result of formulas or cursor pointing do not APPEND correctly.
4. Blank rows are not APPENDed into dBASE III PLUS.
5. APPENDing a 1-2-3 character data type such as "23.33," into a dBASE III PLUS numeric field results in erroneous data.
6. In 1-2-3, make sure that the last cell in a row of data is not empty. If the last cell is empty, the last field of the corresponding dBASE III PLUS record and the first field of the next record will be blank.

If you are using release 2 and want to APPEND it into dBASE III PLUS, use 1-2-3's Translate utility to convert the release 2 (.WK1) file to a release 1A file (.WKS). 1-2-3, release 2 will import a release 1A file without any need for conversion. 1-2-3, version 2 also has a file conversion utility that can export dBASE III database files.

Conclusion

In this article we discussed how data is represented in the DIF, SYLK, and WKS formats. For information on the SDF and DELIMITED file formats, refer to "Sharing Data with Other Programs" in the June 1985 issue of ***TechNotes***.

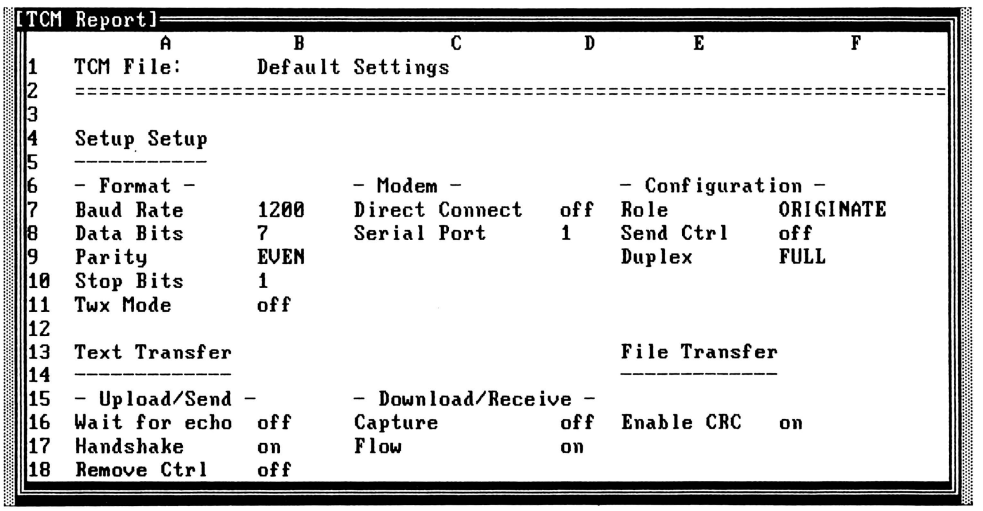
TCM File Report

by Christopher White

If you use telecommunications often, you probably need to document your .TCM file definitions. As you work, you may also want to view your Telecommunications settings all at once rather than navigate through the various menus one at a time. Both of these basic tasks can be met by a spreadsheet that reports the various telecommunications settings using the **@sense** function.

The following program, TCM_builder.FW2, creates and formats a spreadsheet called, TCM Report that you can print with a simple print macro also listed here.

The following figure shows what TCM Report looks like:



[TCM Report]

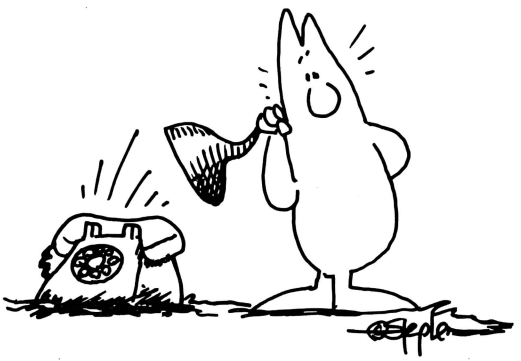
	A	B	C	D	E	F
1	TCM File:	Default Settings				
2	==========	==========	==========	==========	==========	==========
3						
4	Setup Setup					
5	----------					
6	- Format -		- Modem -		- Configuration -	
7	Baud Rate	1200	Direct Connect	off	Role	ORIGINATE
8	Data Bits	7	Serial Port	1	Send Ctrl	off
9	Parity	EVEN			Duplex	FULL
10	Stop Bits	1				
11	Twx Mode	off				
12						
13	Text Transfer				File Transfer	
14	----------				----------	
15	- Upload/Send -		- Download/Receive -			
16	Wait for echo	off	Capture	off	Enable CRC	on
17	Handshake	on	Flow	on		
18	Remove Ctrl	off				

Figure 1.

To set up TCM Report, you must first create TCM_Builder.FW2 and execute it. TCM_Builder creates a blank spreadsheet, formats the columns, places each cell formula into the formula area of each cell with **@setformula**, and moves TCM Report into your Library. To set up TCM_Builder.FW2, create an empty frame on the desktop and name it TCM_Builder. Then enter the formula below into the formula area of the TCM_Builder frame. To create TCM Report, highlight TCM_Builder and press **RECALC (F5)**.

The second portion of building and using TCM Report is Print_report.FW2. This program prints TCM Report from memory with **@print.** To set up Print_report, create an empty frame in the macros section of your Library and enter the formula into the formula area of the frame. Name it with the appropriate **Alt-** < key > combination you want to invoke it with.

To use this system, simply press the **Alt**- < key > you defined Print_report to and TCM Report will print immediately to the printer. If you have not loaded a .TCM file to the desktop, TCM Report will report the default settings and place the string "Default Settings" in place of the .TCM filename at the top of the report.



Program to Generate the Report Spreadheet

```
; TCM_Builder.FW2
;
; by Christopher White
;    November 1, 1986
;
;
; Create spreadsheet.
@performkeys("{Esc}{Esc}{Esc}{Esc}{Ctrl-Uplevel}{Ctrl-C}S"&
             "TCM Report{Return}{F4}{F4}{Return}"),
; Set column alignment.
@performkeys("{Ctrl-N}L"),
; Set print attributes.
@performkeys("{Ctrl-P}FP05{Return}W80{Return}{Esc}"),
; Size Columns.
@performkeys("{Dnlevel}{Ctrl-Home}{F4}{Home}{Ctrl-1}{Ctrl-4}"&
             "{Rightarrow}{Return}"),
@performkeys("{Rightarrow}{F4}{Home}{Ctrl-7}{Rightarrow}"&
             "{Return}"),
@performkeys("{Rightarrow}{F4}{Home}{Ctrl-1}{Ctrl-6}"&
             "{Rightarrow}{Return}"),
@performkeys("{Rightarrow}{F4}{Home}{Ctrl-4}{Rightarrow}{Return}"),
@performkeys("{Rightarrow}{F4}{Home}{Ctrl-1}{Ctrl-2}"&
             "{Rightarrow}{Return}"),
@performkeys("{Rightarrow}{F4}{Home}{Ctrl-1}{Ctrl-3}"&
             "{Rightarrow}{Return}"),
@performkeys("{Ctrl-Home}{Uplevel}"),
; Set formulas.
@setformula(TCM Report.A2,"TCM File:"),
@setformula(TCM Report.A3,@rept("=",75)),
@setformula(TCM Report.A5,"Setup Setup"),
@setformula(TCM Report.A6,"----------"),
@setformula(TCM Report.A7,"- Format -" ),
@setformula(TCM Report.C7,"- Modem -"),
@setformula(TCM Report.E7,"- Configuration -"),
@setformula(TCM Report.A8,"Baud Rate"),
@setformula(TCM Report.B8,@sense(#ATSSFB)),
@setformula(TCM Report.C8,"Direct Connect"),
```

```
@setformula(TCM Report.D8,@if(@sense(#ATSSMD),"on","off")),
@setformula(TCM Report.E8,"Role"),
@setformula(TCM Report.F8,@select(@sense(#ATSSCR) + 1,
                                  "ORIGINATE","ANSWER")),
@setformula(TCM Report.A9,"Data Bits"),
@setformula(TCM Report.B9,@sense(#ATSSFD)),
@setformula(TCM Report.C9,"Serial Port"),
@setformula(TCM Report.D9,@sense(#ATSSMS)),
@setformula(TCM Report.E9,"Send Ctrl "),
@setformula(TCM Report.F9,@if(@sense(#ATSSCS),"on","off")),
@setformula(TCM Report.A10,"Parity"),
@setformula(TCM Report.B10,@select(@sense(#ATSSFP),"ODD","EVEN")),
@setformula(TCM Report.E10,"Duplex"),
@setformula(TCM Report.F10,@select(@sense(#ATSSCD) + 1,"HALF","FULL")),
@setformula(TCM Report.A11,"Stop Bits"),
@setformula(TCM Report.B11,@sense(#ATSSFS)),
@setformula(TCM Report.A12,"Twx Mode"),
@setformula(TCM Report.B12,@if(@sense(#ATSSFT),"on","off")),
@setformula(TCM Report.A14,"Text Transfer"),
@setformula(TCM Report.A15,"-------------"),
@setformula(TCM Report.E14,"File Transfer"),
@setformula(TCM Report.E15,"-------------"),
@setformula(TCM Report.A16,"- Upload/Send -"),
@setformula(TCM Report.C16,"- Download/Receive -"),
@setformula(TCM Report.A17,"Wait for echo"),
@setformula(TCM Report.B17,@if(@sense(#ATTUW),"on","off")),
@setformula(TCM Report.C17,"Capture"),
@setformula(TCM Report.D17,@if(@sense(#ATTDC),"on","off")),
@setformula(TCM Report.E17,"Enable CRC"),
@setformula(TCM Report.F17,@if(@sense(#ATFE),"on","off")),
@setformula(TCM Report.A18,"Handshake"),
@setformula(TCM Report.B18,@if(@sense(#ATTUH),"on","off")),
@setformula(TCM Report.C18,"Flow"),
@setformula(TCM Report.D18,@if(@sense(#ATTDF),"on","off")),
@setformula(TCM Report.A19,"Remove Ctrl"),
@setformula(TCM Report.B19,@if(@sense(#ATTUR),"on","off")),
@setformula(TCM Report.A22,"Setup"),
@setformula(TCM Report.A23,"-----"),
@setformula(TCM Report.A24,"Emulation"),
@setformula(TCM Report.B24,@if(@sense(#ATSE),"on","off")),
@setformula(TCM Report.A25,"Terminal"),
@setformula(TCM Report.B25,@select(@sense(#ATST) + 1,
                                   "DEC VT-100",
                                   "Televideo 920",
                                   "IBM 3101",
                                   "LSI ADM 3A",
                                   "ADDS Viewpoint"
                                  )
),
@TCM Report,
; Put it in the Library.
@performkeys("{F7}{Scroll-Lock}{Home}{Dnlevel}{Ctrl-Home}"&
             "{Ctrl-End}{Return}"),
@performkeys("{Ctrl-Uplevel}")
```

Program to Send the Report to the Printer

```
; Print_report.FW2
;
; by Christoher White
;    November 1, 1986
;
@local(tcmfile),
@performkeys("{Ctrl-A}TSL{F6}{End}{Return}"),
tcmfile := @textselection,
@if(@iserr(@mid(tcmfile,1,1)),
   tcmfile := "Default Settings"
),
@performkeys("{Esc}{Esc}"),
TCM Report.b2 := tcmfile,
@TCM Report,
@print(TCM Report)
```

A Simple Data-Entry Program for Spreadsheets and Databases

by Karen Robinson and Christopher White

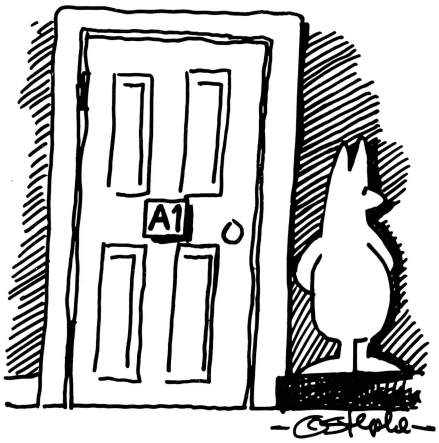
This article is the first installment of a two-part discussion on entering data into Framework II spreadsheets and databases. Part One introduces the basics and offers a simple program for automating data entry. Part Two will discuss programming techniques and will offer an improved version of the program.

Data entry is a fundamental task for anyone using spreadsheets and databases for data management. In Framework II, you can enter data interactively or by automating the process with a FRED macro or program. Both methods have advantages. The interactive method allows you to work informally requiring little or no set-up time, a useful approach for small or seldom-performed tasks. If you have a large, repetitive tasks, you should consider automated data entry to increase productivity. Framework II provides a number of resources that can assist you in creating automated data-entry procedures. The drawback of automation lies in the amount of development time required to create and debug data-entry macros. Fortunately, the process is not as time-consuming as it may seem at first.

Interactive Data Entry

In Framework II, interactive data entry is fast and simple to set up. Position the cursor on the spreadsheet cell or database field into which you want to enter data, type the cell's contents, and press **Return** or **Tab** to terminate the entry. Framework II determines the data type and puts you into formula or label edit appropriately. Blatant errors, such as attempts to enter mixed data types into formula edit, are trapped and return an error constant. For example, an error will result if you attempt to enter a street address that begins with the street number and ends with the street name without first pressing the **Spacebar**.

The convenience of this automatic data typing is offset by the problem of mixed data types. You can deal with this situation by pressing either **EDIT FORMULA (F2)** or **Spacebar** to enter the proper edit area for the data type you are entering. Remembering to do this can be tedious if you have a number of cells or fields to enter. A macro that senses the mixed data type and selects the proper mode therefore, can provide improved productivity.



A macro can help you avoid such subtle errors in other situations. You might, for example, be entering a cell value as numeric when all of the previous entries for this column or field were entered as character. This might happen if the field or column you are entering holds a zip code or a telephone number. Or, you might have a range or set of acceptable values and you want to prevent the entry of unacceptable values. Or, you might need to control the format of the column or field and it differs from the form when you input it. For example, you can enter a date interactively as character type and in a variety of ways: September 12, 1986 or 09/12/86. If, however, you need to perform a calculation with the column or sort it in date order, you must use the **@date** function to enter a date type into the cell and **@date** requires you to explicitly specify the year, month, and day in that order as numeric arguments of the function. As you can see, entering a date type can be tedious if any repetition is involved. In general, a simple data-entry macro can help you to work around these small difficulties increasing your work pace and reducing the probability of error.

Data Entry under Program Control

Like any programming language, FRED offers a number of ways to structure a data-entry program depending on the needs of the user and the style of the programmer. Logically, the basic structure of this type of program can be divided into three parts:

1. Main driver
2. Record entry
3. Field entry

In general, the main driver of the program sets up the working environment. This includes selecting the appropriate file or frame, initializing memory variables, and then allowing you to enter records until you press an escape key to terminate the append operation. At the record level, the program moves the cursor to the next blank record and executes an entry routine for each field. The field-level routine gets the field contents from you in much the same manner as you would interactively. It may also include error-checking and formatting of the field.

The best way to set up this type of program is to break it up into small modules so that it is easy to debug and to change. Add.FW2, the program presented here, demonstrates this idea by separating the input formulas for each field into separate frames. It then directs data entry by moving the cursor from field to field automatically. You first move the cursor to an empty record and then engage the macro with an **Alt**-< key > combination. You are then prompted to enter information for the first field. When you press **Return**, the cursor advances to the next field. After you enter the last field of the record, the cursor moves to the first field of the next blank record. To terminate the Add.FW2, press **Esc**.

Add.FW2 works by calling the field routines one at a time from Getfields. Within each field routine, it tests for **Esc**. If you pressed **Esc**, **@return** exits the routine and returns the value 0 as the result of the field routine. In Getfields, the value of the field routine is tested. If it is 0, then the program terminates; otherwise, the next field routine is executed.

Setup

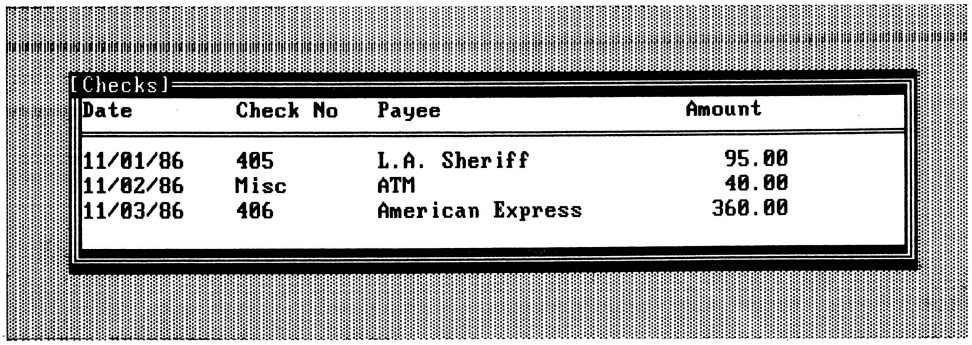
In order to keep Add.FW2 simple, the following features are not implemented but will be discussed and presented in Part Two of the article:

1. Error or special range checking.
2. Format of data when inputting. You are not forced to enter data in any special format. Add.FW2 does, however, use the **@inputline** function, which allows you to display a default value for each field.

In its current form, Add.FW2 makes the following assumptions about your data-entry environment:

1. The cursor is located in a spreadsheet or database.
2. The spreadsheet or database contains enough blank records or rows for the current entry session.
3. The cursor is placed on the first blank row that you want to enter data into.
4. The data will be entered into contiguous columns, and the first column is the left-most column in the data object.
5. Database frames are in Table View.

Add.FW2 can be used with spreadsheets and databases. It uses this sample four-column spreadsheet:



[Checks]

Date	Check No	Payee	Amount
11/01/86	405	L.A. Sheriff	95.00
11/02/86	Misc	ATM	40.00
11/03/86	406	American Express	360.00

To set up the Add.FW2, follow these steps:

1. Create an outline frame with five subframes.
2. Label the containing frame Add. Label the subframes Getfields, Trans date, Checkno, Payee, and Amount.
3. Enter the formulas below into the formula area of each respective frame by pressing **EDIT FORMULA (F2)** and the **ZOOM (F9)**.

```
; Add.FW2
;
; by Christopher White
;    November 1, 1986
;
;
@[{Alt-A}].Getfields
; Getfields.FW2
;
;
@eraseprompt,
@prompt("Press Esc to quit",32),
@while(#TRUE,
       @if(@Trans date = 0, @return(0)),
       @if(@Checkno = 0, @return(0)),
       @if(@Payee = 0, @return(0)),
       @if(@Amount = 0, @return(0)),
)
```

```
; Trans date.FW2
;
;
@local(date),
date := @decimal(@date),
; Entry formula for transaction date.
Trans date:= @inputline("Enter transaction date",date,#NO,#YES,),
; Test for escape key.
@if(@key = {Esc}, @return(0)),
```

```
; Place the data into the current cell.
@performkeys(" "&Trans date&"{Tab}"),
; Return OK flag.
@return(1)
```

```
; Checkno.FW2
;
;
; Enter check number.
checkno := @inputline("Enter check number",,#NO,#YES,#YES),
; Test for escape key.
@if(@key = {Esc},@return(0)),
@performkeys(" "&checkno&"{Tab}"),
; Return OK flag.
@return(1)
```

```
; Payee.FW2
;
;
; Enter payee name.
payee := @inputline("Enter Payee",,#NO,#YES,#YES),
; Test for escape key.
@if(@key = {Esc},@return(0)),
@performkeys(" "&payee&"{Tab}"),
; Return OK flag.
@return(1)
```

```
; Amount.FW2
;
;
; Enter amount of check.
amount:= @inputline("Enter amount","0.00",#NO,#YES,#YES),
; Test for escape key.
@if(@key = {Esc},@return(0)),
; Amount is entered as numeric.
@performkeys("{F2}"&amount&"{Return}{Dnarrow}{Home}"),
; Return OK flag.
@return(1)
```

Conclusion

Once you have typed the program, set it up as a macro: Move the frame into the macro section of the Library or use the **@setmacro** command in the formula area of the containing frame, Add. If you move it into the macro section of the Library, rename Add to "{Alt-A} ; Data-entry program" and enter the formula "@[{Alt-A}].Getfields" into its formula area. If you want the program to reside on the desktop, type the following **@setmacro** command line into the formula area of the containing frame, Add:

```
@setmacro({Alt-A},Add.Getfields)
```

The advantage of using the Library frame is that you do not have to load the macro from disk or **RECALC (F5)** before using the program.

To execute this data-entry program, press **RECALC (F5)** (if it does not reside in the Library frame), move the cursor to the first column of a blank record in the spreadsheet, press **Alt-A**, and follow the screen prompts to enter records. Press **Esc** to exit the program.

Next month, we will present the second article in this two-part series. We will describe how to enhance this simple data-entry macro to do data validation, select the field choice from a list, and create dependent fields.

USAGE TIPS

dBASE III PLUS

by Joseph Perret, J. Mike Hedblom, and Christopher White

The dBASE III PLUS database file header is one byte shorter than that created by all other versions of dBASE III. Earlier versions put an "extra" byte (00H) between the header and the beginning of the data. dBASE III PLUS no longer includes this byte and the data begins immediately following the last byte of the header (0DH). The number of bytes in the header (bytes 8 and 9) reflect the smaller header size.

dBASE III PLUS has no problem reading files from earlier versions as long as bytes 8 and 9 correctly reflect the actual size of the header.

This procedure will create a file one byte smaller than the original file:

```
* ---This is a 1.1 created file.
USE Test
COPY TO Test1
DIR                 && Test1 is one byte smaller than Test.
```

.DBF File Size

The algorithm for figuring the size of a database file is as follows.

((<# of bytes> + 1) * <# of records>) + (<# of fields> * 32) + 34

where

bytes :: = total bytes of fields in each record
records :: = number of records in database file
fields :: = number of fields in database file

CREATE FROM

In dBASE III PLUS, the maximum column width that CREATE/MODIFY REPORT defaults to is 30 characters. If an expression result is longer than 30 characters, you must change the column width.

Error Messages by Number

The following is a list of dBASE III PLUS error messages in numeric order.

1 File does not exist.
2 Unassigned file no.
3 File is already open.
4 End-of-file encountered.
5 Record is out of range.
6 Too many files are open.
7 File already exists.
8 Unbalanced parenthesis.
9 Data type mismatch.
10 Syntax error.
11 Invalid function argument.
12 Variable not found.
13 ALIAS not found.
14 No find.
15 Not a dBASE database.
16 *** Unrecognized command verb.
17 Cannot select requested database.
18 Line exceeds maximum of 254 characters.
19 Index file does not match database.
20 Record is not in index.
21 Out of memory variable memory.
22 Out of memory variable slots.
23 Index is too big (100 char maximum).
24 ALIAS name already in use.
25 Record not inserted.
26 Database is not indexed.
27 Not a numeric expression.
28 Too many indices.
29 File is not accessible.
30 Position is off the screen.
31 Invalid function name.
33 Structure invalid.
34 Operation with Memo field invalid.
35 Unterminated string.
36 Unrecognized phrase/keyword in command.
37 Not a Logical expression.
38 Beginning of file encountered.
39 Numeric overflow (data was lost).
41 .DBT file cannot be opened.
42 CONTINUE without LOCATE.
43 Insufficient memory.
44 Cyclic relation.
45 Not a character expression.
46 Illegal value.
47 No fields to process.
48 Field not found.
50 Report file invalid.
51 End of file or error on keyboard input.

52 No database in USE.
53 There are no files of the type requested in this drive or catalog.
54 Label file invalid.
55 Memory Variable file is invalid.
56 Disk full when writing file:
57 ***Execution error on CHR() : Out of range.
58 ***Execution error on LOG() : Zero or negative.
59 ***Execution error on SPACE() : Too large.
60 ***Execution error on SPACE() : Negative.
61 ***Execution error on SQRT() : Negative.
62 ***Execution error on SUBSTR() : Start point out of range.
63 ***Execution error on STR() : Out of range.
65 Internal error : Unknown command code:
66 Internal error : CMDSET():
67 Internal error : EVAL work area overflow
68 Internal error : Illegal opcode.
70 ** WARNING ** Data will probably be lost. Confirm (Y/N).
72 ALTERNATE could not be opened.
73 ^^ Expected ON or OFF.
74 ^—- Truncated.
75 ^—- Out of range.
76 ***Execution error on - : Concatenation string too large.
77 ***Execution error on + : Concatenation string too large.
78 ***Execution error on ^ or ** : Negative base, fractional exponent.
79 ***Execution error on STORE : String too large.
81 Invalid date.
82 ** Not Found **
86 ^— Keyword not found.
87 ***Execution error on NDX() : invalid index number.
88 ***Execution error on REPLICATE() : String too large.
89 Cannot erase a file which is open.
90 Operation with Logical field invalid.
91 File was not LOADed.
92 Unable to load COMMAND.COM.
93 No PARAMETER statement found.
94 Wrong number of parameters.
95 Valid only in programs.
96 Mismatched DO WHILE and ENDDO.
99 Invalid DOS SET option.
101 Not suspended.
102 ***Execution error on STUFF() : String too large
103 DOs nested too deep.
104 Unknown function key.
105 Table is full.
106 Invalid index number.
107 Invalid operator.
108 File is in use by another.
109 Record is in use by another.
110 Exclusive open of file is required.
111 Cannot write to a read-only file.
112 Index expression is too big (220 char maximum).
113 Index interrupted. Index will be deleted if not completed.
114 Index damaged. REINDEX should be done before using data.
115 Invalid DIF File Header.

116 Invalid DIF vector - DBF field mismatch.
117 Invalid DIF Type Indicator.
118 Invalid DIF character.
119 Invalid SYLK File Header.
120 Invalid SYLK Dimension Bounds
121 Invalid SYLK File Format.
122 Data Catalog has not been established.
123 Invalid printer port.
124 Invalid printer redirection.
125 Printer not ready.
126 Printer is either not connected or turned off.
127 Not a valid VIEW file.
128 Unable to SKIP.
129 Unable to LOCK.
130 Record is not locked.
131 Database is encrypted.
132 Unauthorized login.
133 Unauthorized access level.
134 Not a QUERY file.
136 Unsupported path given.
137 Maximum record length exceeded.
138 No fields were found to copy.
139 Cannot JOIN a file with itself.
140 Not a PFS file.
141 Fields list too complicated.
142 Relation record is in use by others.
143 Query not valid for this environment.
146 Maximum path length exceeded.
147 Cannot append in column order.
148 Network server busy.
149 Master catalog is empty.

Note also that the following error is not documented.
146 Maximum path length exceeded.

INDEXing Street Addresses

If you have a name and address database file, you might need to INDEX your database file in street order. The basic requirement is to create an index file that keeps all like streets together, ordering the addresses by numeric value within each street. The reason for this is that INDEXing ON the street alone will not give you a very meaningful order. An additional requirement is to assign all blank and "P.O. Box" addresses a value that groups them at the top or bottom of the index so that you can easily skip over them.

The following commands use IIF() in combination with other string functions to create an index key that will produce an index file in street order. When you use these commands, enter the assignments to the memory variables "ae" and "af" first, and then enter the INDEX command on one continuous command line. As you type the INDEX command line, be sure to ignore embedded blanks and semicolons within the INDEX expression to keep the expression length less than the 220-character limit that dBASE III PLUS imposes.

```
* ---INDEXing street names.
ae = [SUBSTR(Street, 1, AT(" ",Street) - 1)]
af = [SUBSTR(Street, AT(" ",Street) + 1, 5)]
INDEX ON IIF( "BOX" $ &ae.;
              .OR. "P.O." $ &ae.;
              .OR. LEN(TRIM(Street)) = 0,;
              " ",;
              &af. + RIGHT(STR(100000 + VAL(&ae),6),5);
         );
      TO Address
```

READKEY()

If you terminate a full-screen operation in dBASE III PLUS with a **Ctrl-W** or **Ctrl-End**, READKEY() always returns 270, regardless of whether or not the contents were altered. In the Developer's Release, READKEY() returned a value of 14 if the field or variable was not changed. Additionally, READKEY() always returns a value of 12 when you terminate a full-screen operation with **Esc** or **Ctrl-Q**.

Semicolon

In dBASE III PLUS, you cannot continue commands issued at the dot prompt with a semicolon. The dBASE III PLUS command line is a 254-character horizontal scroll space. Previous versions of dBASE III allowed you to continue a dot prompt command line with a semicolon.

SET CATALOG ON

If you assign an ALIAS when you USE a database file, the catalog does not store the ALIAS but instead stores the filename as the ALIAS. For example,

```
SET CATALOG TO <filename>
USE Test1 ALIAS Testfile
SELECT 10
LIST
  ↑___ This show the ALIAS as "Test1,"
       not "Testfile."
```

SET COLOR TO

In dBASE III PLUS, the command,

SET COLOR TO N, <**Return**>

produces black on black, even though there is no space after the comma.

"***Execution error on SUBSTR(): Start point out of range."

If you are using the SUBSTR( ) function with memory variables or with an expression whose value might be null or shorter in length than the starting position argument of the SUBSTR( ) function, you will get the error message "***Execution error on SUBSTR( ): Start point out of range." This is not a terminal error condition as is the case with a syntax error. The SUBSTR( ) function terminates gracefully, returning a null as the result. The execution error, however, does present a problem. If SET CONSOLE is ON and the execution error occurs, the error message will echo to the screen, disturbing your display. If SET PRINT is ON or the substring expression is part of any command that is sending output with the TO PRINT clause, the error message will echo to the printer. This is particularly a problem with REPORT and LABEL FORMs.

There are three strategies you can use to suppress the display of the error message. The use of each depends on what commands are being used with the substring expression.

1. If you are assigning memory variables or using @...SAY to send the result of the substring expression to the screen or printer, you can SET CONSOLE OFF to suppress the error message.

2. If you are using a TO PRINT command such as LABEL FORM TO PRINT, use the IIF( ) function as the starting position expression to return a zero if the length of the character string is less than the start position you really want. SUBSTR( ) always returns a null string if the starting position is zero. For example, the following expression returns a null string if the length of the memory variable "x" is less than three.

   ```
   SUBSTR(x, IIF(LEN(x) < 3, 0, 3), 6)
   ```

   Use a SUBSTR( ) with this form to suppress the execution error message.

3. Lastly, you can use ON ERROR to trap execution errors. Define the argument of the ON ERROR condition as a dummy expression such as,

   ```
   ON ERROR dummy = 0
   ```

   before you execute a command that causes the execution error. Be aware that this approach is limited since you may need the ON ERROR condition for other purposes. In addition, branching to a command file in order to trap a range of error conditions terminates REPORT or LABEL FORMs.

USE

If byte 15 of a .DBF file header contains a value other than a null, USEing the file in dBASE III PLUS produces the error message "Database is encrypted." This byte is used by the dBASE Administrator to mark a file as encrypted. Earlier versions of dBASE III ignored this byte. If you encounter this error message when USEing a database file in single-user mode, run NEWHEAD.BAS or use DEBUG to change the byte back to a null. Note that in order to use DEBUG, your database file must be smaller in size than the amount of free memory available prior to running DEBUG.

Framework II

by Chris Lo Vasco and Christopher White

Aligning Numbers in Database Forms View

Sometimes you want to print a database in Forms View with numeric fields aligned on the decimal point. In Framework II, the default display for data in Forms View is left-justified so that so the decimal points are not aligned. The following illustrates a method to right-justify the display of numeric fields in Forms View to get decimal point alignment.

First, while in Table View, **EXTEND SELECT (F6)** the entire field column to be right-justified. Next, press **VIEW (F10)** to toggle to Forms View and **then SIZE (F4)** the field to the desired width. Finally, from the **Words** menu, set the left margin to zero, the right margin equal to the width of the column, and the **Flush Right** menu option ON. Repeat these steps for each field you wish to right-justify.

Note that if you want to stack numeric fields in a column, one on top of another, format each numeric field with the same settings from the **Numbers** menu, **SIZE (F4)** each field to the same width, and set the left and right margin values in the **Words** menu to the same settings.

Range Search and Replace in a Word Frame

Occasionally, you may want to search and replace a string in a highlighted area of a word frame instead of the entire frame contents. You might want to do this if you are writing or editing programs, and you want to change a keyword in a block rather than in an entire program or module. Framework II supports search and replace of selected regions in spreadsheet and database frames but not in word frames. To be able to do this, you need a macro. The following macro, Replace.FW2, allows you to search and replace selected or highlighted areas of a word frame.

It works first, by prompting you for the string to search for and the string to replace it with. After you enter these, it cuts the currently highlighted text to the Cut Paste Buffer with the Cut macro, **Alt-F7**, and then performs the search-and-replace operation from the Locate pull-down menu with your strings as arguments. Once the replace operation is complete, Replace pastes the selected text from the Cut Paste Buffer back to its original position in your frame and repaints the screen. Unlike the **Locate:Replace** menu option, Replace can only globally search and replace all occurrences of the search string. If the **Locate:Search** finds no matches, Replace uses the **@isabend** function to detect this condition and act appropriately. You may want to note this usage of **@isabend** especially when used in combination with either the Search or Replace options of the **Locate** pull-down menu.

To set up Replace.FW2, create an empty frame in the Macros section of your Library, name it with the appropriate **Alt-** < key > combination, press **EDIT FORMULA (F2)** and **ZOOM (F9)**, and enter the following into the formula area of the frame. To execute Replace, highlight the block of text you want to search and replace within and press the **Alt-** < key > .

```
; Replace.FW2
;
; by Christopher White
;    November 1, 1986
;
@local(locate_for, replace_with),
; Get the search and replace arguments.
locate_for := @inputline("Search for..."),
replace_with := @inputline("Replace with..."),
@echo(#OFF),
; Cut.
@performkeys("{Alt-F7}"),
; Go to the paste buffer.
@performkeys("{Ctrl-L}GCut Paste Buffer{Return}{F9}{Ctrl-Home}"),
; Replace all occurrences.
@isabend(#FALSE),
@performkeys("{Ctrl-L}R" & locate_for & "{Return}" &
                           replace_with & "{Return}{End}"),
@if(@isabend = #TRUE,
     ; There were no replacements.
     @list(@performkeys("{Esc}"),
           @performkeys("{Ctrl-L}G{Return}"),
     ),
     ; There was at least one replacement.
     @list(@performkeys("{Ctrl-L}G{Return}"),
           @performkeys("{Del}{Alt-F8}{F9}")
     )
),
@echo(#ON)
```

Word Wrap a Spreadsheet Label

If you have been a 1-2-3 user, you may have used the /Range Justify command to word wrap long spreadsheet cell labels. Probably, you wish Framework II supported this feature. Although Framework II can word wrap long labels if you **ZOOM (F9)** a spreadsheet label, this does not word wrap within a column as 1-2-3's /Range Justify does. It instead word wraps the cell label within the label space for that label only.

The following macro, Wrap_cell.FW2, word wraps long spreadsheet labels much as /Range Justify does, wrapping a label down a column and within the column width. For example, Figure 1 below shows a long label in the spreadsheet cell Spreadsheet.A1 before Wrap_cell was executed on it. Figure 2 shows the result of executing Wrap_cell on Spreadsheet.A1 within the column width of column A.

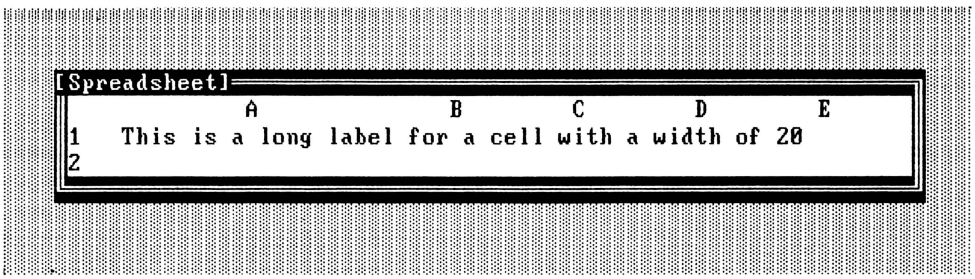


Figure 1.

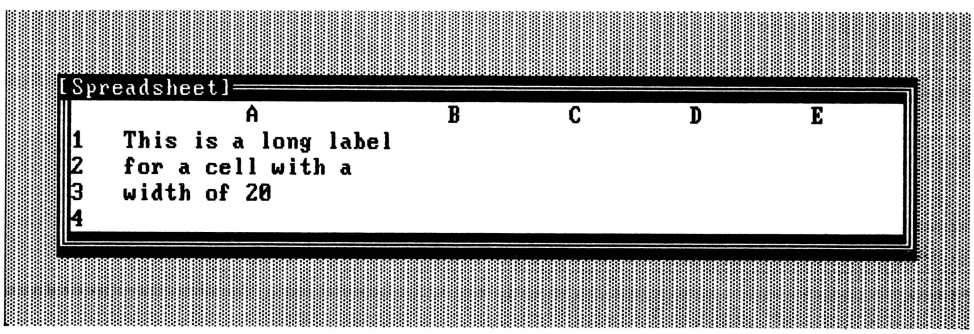


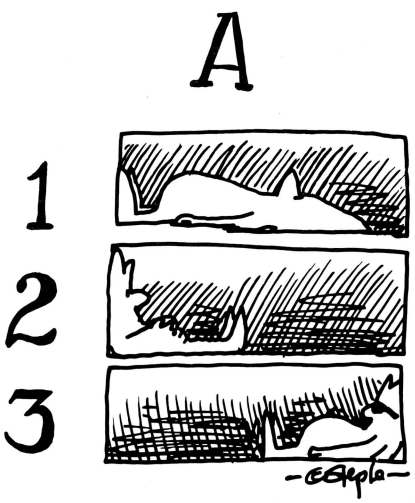
Figure 2.

Wrap_cell first cuts the cell label to a temporary frame and then sets the margin of the temporary frame to the width of the spreadsheet cell. Setting the right margin automatically reforms the label to the width of the original spreadsheet cell width. Wrap_cell then pastes the wrapped text back to the spreadsheet line by line.

To set up Wrap_cell.FW2, create an empty frame in the Macros section of your Library, name it with the appropriate **Alt**-< key > combination, and enter the following formula into the formula area of the frame. To execute the macro, highlight the spreadsheet cell that has the long label and press the **Alt**-< key > for Wrap_cell.

As you work with this macro, note that it has limitations. First, Wrap_cell does not check for existing labels or formulas before writing to a new cell in the column. This means that it will overwrite cells directly below the cell with the long label. Second, Wrap_cell cannot deal with protected cells. If it attempts to write to a protected cell, at the very least you will get an ambiguous and possibly an unusable result. Last, the results of Wrap_cell cannot be undone. If you don't like the result, you will have to reconstruct the original long label manually. To avoid these potential problems, use the following guidelines:

1. Before beginning work, be sure there are no protected cells in the spreadsheet by executing the **Locate:Allow Editing** from the frame border of the spreadsheet.
2. Construct your spreadsheet from left to right and top to bottom. This will prevent overwriting existing labels and formulas.
3. As you work on each column for the first time, size it appropriately so that when you wrap long labels you will get the correct wrapping margin.




```
; Wrap_cell.FW2
;
; by Christopher White
;    November 1, 1986
;
@local(cellwidth, cellname, originalcell, errorbeep),
errorbeep := "@beep(300,1),@beep(499,5),@beep(700,5),
              @beep(300,1),@beep(499,5),@beep(700,5),
              @beep(300,1),@beep(499,5),@beep(700,5)",
; Get cell name.
cellname := @setselection,
originalcell := cellname,
; Check for frame type and location.
@if(@frametype <> 11,
    @return(
        @list(
            @errorbeep,
            @prompt("You are not in a spreadsheet, "&
                    "press any key to continue..."),
            @nextkey,
            @eraseprompt
        )
    )
),
; Check for spreadsheet name.
@if(@mid(originalcell,1,2) = "[]",
    @return(
         @list(
             @errorbeep,
             @prompt("Spreadsheet has no name, "&
                     "press any key to continue..."),
```

```
            @nextkey,
            @eraseprompt
)
     )
   ),
; Get its size for word-wrap parameter.
@performkeys("{F4}"),
cellwidth := @panel1,
@eraseprompt,
@prompt("Please wait ...",35),
@echo(#OFF),

; Cut the text to a temporary buffer and wrap to cell width.
@performkeys("{Return} {F6}{Ctrl-End}{Alt-F7}{Del}{Return}{Ctrl-Uplevel}"),
@performkeys("{Ctrl-C}E Temp{Return}{Dnlevel}{Alt-F8}{Ctrl-W}R"&
             @integer(cellwidth)&"{Return}"),
; Move stuff back to the spreadsheet.
@performkeys("{Ctrl-Home}{F6}{End}{Alt-F7}{Del}"),
@setselection(cellname),
@performkeys(" {Alt-F8}{Uplevel}{Dnarrow}"),
@isabend(#FALSE),
@while(@isabend <> #TRUE,
     cellname := @setselection("Temp"),
     @performkeys("{Dnlevel}{Ctrl-Home}{F6}{End}{Alt-F7}{Del}"),
     @if(@isabend = #TRUE,

          ; Clean up and quit.
          @list(
               @performkeys("{Uplevel}{Del}"),
               @setselection(originalcell),
               @echo(#ON),
               @return(0)
          ),

          ; Go to the next cell and paste.
          @list(
               @setselection(cellname),
               @performkeys(" " & "{Alt-F8}{Uplevel}{Dnarrow}")

          )
     )
)
```

ANOMALIES

We use the word "anomaly" to mean any one of the following:

- *INCORRECT COMMAND (or function) IMPLEMENTATION. A command or function does not execute as specified.*
- *INCOMPLETE COMMAND (or function) IMPLEMENTATION. A command or function does not execute as specified in every use of the command or function.*
- *DATA OR LOGICAL BOUNDARY LIMITATION. A command or function cannot perform an operation at a specified data or logical boundary.*

  NOTE: Ashton-Tate reserves the right to withhold any further information that may reveal proprietary or trade secret information.

Framework II

DOS Access with DOS 3.2

If you run the PC-DOS version 3.2 of FORMAT.COM from within DOS Access without doing a DOS CLS, a display problem occurs. When the screen output from FORMAT.COM echos in the DOS Access frame, it updates the screen continually at the same cursor location overwriting existing information.

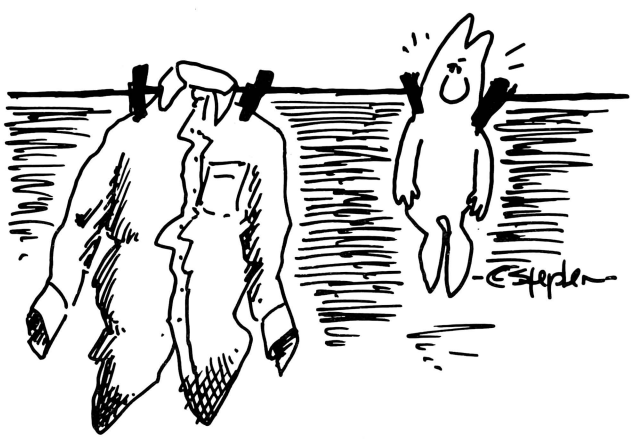
To work-around, issue a DOS CLS to clear the screen and place the display in full-screen mode before executing FORMAT.COM.

ON-LINE SERVICES

Do you have a modem? Do you subscribe to CompuServe or The Source? If so, Ashton-Tate offers technical support on all of these electronic information services. You can quickly access the kind of quality technical information that you have come to expect from ***TechNotes***, and you can do it through your computer system. To take advantage of this progressive new support method, you must subscribe to one of the services listed below. There are no additional charges for using our on-line support services, apart from the normal connect-time fees for each service. You can get specific subscription information by calling one of the following numbers:

Service	Subscriber Information	Command to find Ashton-Tate
CompuServe	(800) 848-8199	GO ASHTON
The Source	(800) 336-3366	ATSIG

To give you an idea of support that is happening on-line, ***TechNotes*** will be publishing regular excerpts from the Ashton-Tate Forum on CompuServe. This monthly material comes from messages our Forum users have posted on the bulletin board. All message headers have been removed to protect the identities of our users. Every other message within a single conversation topic has been italicized to enhance readability.



dBASE II, dBASE III, dBASE III PLUS:

Have you ever run across the problem where one user has a database file open (not EXCLUSIVE) but with one record locked with RLOCK(), and another user needs to APPEND a record? I have currently got my setup to automatically kick over to a temporary database file and APPEND the new records back into the original later, but I don't like that solution. Have you heard of any better ones?

I test for FLOCK() before doing an APPEND BLANK. Then, if the database file cannot be locked, control is passed into a RETRY loop. If the record is locked too long for this approach to work, you may have to READ memory variables and then LOCK/REPLACE/UNLOCK so that the locked time is decreased to a minimum.

.....

Can anyone give me a sketch of the code needed to allow a user to page back and forth between multiple menu pages before making a choice? I have an application that has about 50 reports in it and would like to let the user roam through several pages before choosing. I can see some kludgey ways to do it, but I suspect that there is some neat, clean, and simple way that I'm just missing.

I've been using READKEY() or INKEY() to catch the ***PgUp*** *and* ***PgDn*** *keys. I increment/decrement a memory variable "screen_num" taking the MOD() of it by the number of screens and then make it a string value to use it as a macro, such as:*

```
DO Format&screen_num. && The screen formats are Format1.PRG,
                      && Format2.PRG, etc.
```

.....

I have SET BELL OFF for an application, but would like it to sound after an error has been trapped and correct input requested to indicate that an error has occurred. Adding CHR(7) to the @..SAY statement only causes ASCII 7 to appear. Is there anyway around this? I am using a Compaq 286. Is the graphics display the problem?

?? CHR(7) will do it.

.....

Can anyone tell me how I can replicate various fields from a prior record when entering lists of names and addresses to save additional typing? My copy of Reflex uses the ditto ("), but I haven't been able to find anything similar in the *dBASE III PLUS Reference Manual* or the other two manuals I bought to help learn the system.

The SET CARRY ON command will do exactly what you want.

.....

I am a Ashton-Tate dealer and have heard conflicting reports about dBASE III PLUS. I have heard that it is not totally compatible with the new IBM PC XT2 and AT2 keyboards. If this is true, is there a patch available?

This is not true. dBASE III PLUS has been certified for use with the new XT2 and AT2. By the way, it's not the keyboard that's a problem; it's the new ROMs. Framework II is the product that had the compatibility problem which is easily corrected with the new keyboard driver available in DL3.

.....

Is it possible to change a memo field to blanks or assign a specific value to it within a .PRG file and without operator intervention? Thanks for your help.

No, you cannot assign a value to a memo field. In fact, the only processing you can do from a .PRG is to bring the memo field up for editing or display it to the screen or printer. You could have a function key defined so that if you pressed it in the memo edit mode, it would perform a series of keystrokes like a macro. But this would not be completely automated.

.....

Can you please tell me if there is an equivalent command in dBASE III PLUS for NOEJECT? I am trying to write an output program that uses continuous forms and everything is fine except that I get a page eject every time I print a new form. Since they are continuous, I waste a form for every one that I print. I am using @ 1,1 SAY < whatever > . I have also tried issuing an @ 0,0 before SET DEVICE TO PRINT. Any assistance you may be able to provide would be greatly appreciated.

You should be able to print forms without losing them on a one-for-one basis. The formfeed is caused by issuing a line number lower than a previously issued one. This causes dBASE III PLUS to automatically issue a formfeed and can be used to your advantage once you get used to it.

I use the following approach,

1. *Initialize the printer with an @ 0,0 SAY < whatever > , setting the form length, etc.*
2. *Always start printing the actual form on line 1 or greater.*
3. *If necessary, issue an EJECT after each form. This isn't necessary as mentioned above, but it is critical to issue an EJECT after the last form. Otherwise dBASE III PLUS's internal counter never gets reset and it will automatically issue a formfeed the next time you use an @ 0, 0.*

.....

I am using the REPORT FORM command. The numbers I am using are in the ten and hundred thousands. I would like to display them with commas. I could use the TRANSFORM( ) function, but that seems to turn the numbers into character data which won't allow me to total and subtotal. I could write a .PRG file and use the PICTURE clause to generate the report, but the REPORT FORM is perfect except for the lack of commas to promote readability of large numbers. Does anyone know a way to punctuate large numbers and total them in a REPORT FORM? Thanks.

Unfortunately, that issue was not addressed in dBASE III PLUS. The only way I know of to do it is to write a .PRG to do a custom report.

.....

I would appreciate your help with one question. My wife uses dBASE III PLUS also and I want to reduce the number of commands that she will need to use. Can the COLORs and INTENSITY be SET as a default? I am not a programmer, but I can change an AUTOEXEC.BAT file if need be. Thanks again.

You can set the defualts in The CONFIG.DB. Use the syntax:

COLOR = < your color sequence >

in place of

SET COLOR TO < your color sequence >
in dBASE III PLUS.

.....

Framework II:

With Framework II in Telecommunications I am communicating with a device that sends lines longer than 80 characters. This causes the screen to scroll sideways. This is very hard to read! Is there anyway to have the frame wrap at 80 characters and not scroll sideways? If I set the right margin to 80, it resets itself to 255. I appreciate your time, thank you.

1. *Unfortunately, 255 characters is always the right margin of the active Telecomm frame. At this time, there is no way to change it. Is there no way to have your host set up for 80 column monitors, so that it sends CR/LFs back to you at the right column? This might be one solution to the problem.*
2. *If you use Terminal Emulation, your screen will wrap at the right edge. If you're in a Telecomm frame on the desktop, however, it will scroll horizontally until either 255 characters are received or a CR/LF is received.*
3. *The only way to force wrapping is to use Terminal Emulation. Try any one of the terminal types.*

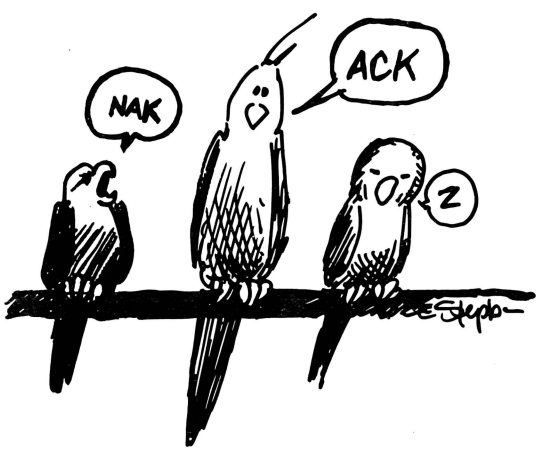
.....

I'm using Framework II on a dual-floppy system. By the time I've copied COMMAND.COM (DOS 2.1) onto System Disk #2 so I can access DOS without having to swap disks, I've just about filled up the disk. I now have several user-defined functions, macros, abbreviations, and printing templates that I'd like to add to my Library.FW2 file, but I can't because there isn't enough space on System Disk #2. The same applies for telecommunications macros.

Until I can get a hard disk, can you recommend what files on System Disk #2 I might be able to delete or move to free enough space to expand my Library? I noticed, for example, that the Help files are split up into HELP1.DAT, HELP2.DAT, and HELP3.DAT. Which .DAT file is used for what? In other words, can one or even two of the HELP.DAT files be deleted and still provide me with assistance in some areas, or do all three have to be present for HELP to work at all? I've also considered moving the .TCM files to the disk in Drive B, but that would mean having to copy them to every one of my data disks, so that isn't very practical. Any suggestions will be appreciated.

Try getting rid of the sample programs, some of the .FW2 files. If you don't use the Disk:Import or Disk:Export now you can dump the Ix.FW2 and Ex.FW2 files also. If you have the help at all, you should have all of the files on disk.

.....



I need a program to find and post to the next empty (last) row on a spreadsheet in a containing frame. The important part looks like this:

```
Str : = "@get(" & Path & ".A1:" & Path & ".A100)",
@while(@not(@isna(@Str)),
      ; Line 1.
      Str : = "@next(" & Path & ".A1:" & Path & ".A35)",
      ; Line 2.
      @Str
      ),
Str : = "@put(" & Path & ".A1:" & Path & ".A100,""Text"")",
@Str
```

These lines work fine except that they skip the first empty row and the result of the **@put** goes in the next one down, but when I replace lines noted above as 1 and 2 with:

```
@next(ExternalFrame.Spreadsheet.A1:
      ExternalFrame.Spreadsheet.A100)
```

it does not skip. I need the path name to the right cell to be a global variable. Can you help?

*That looks painful. Can you do it with **@performkeys**, instead of region walking? If you do a* **Ctrl-Dnarrow** *and then **Dnarrow**, you will be at the next empty row. Then, I'd perform a series of **Rightarrows** across, plopping the data into each cell with **@performkeys**. Otherwise, you will find building region-walking references that are constantly changing can give you a headache. Another problem you are having is that you are referencing two different regions in your **@get** and your **@next**. They are not working on the same region. If they were both working on A1:A100, you'd be much better off.*

Your Ctrl-Dnarrow and Dnarrow solution is brilliant and simple. I wonder why didn't I think of that. About referencing two different regions, it was a typo in my message.

.....

I've got the AST 6-Pak Premium with 256K on the motherboard of my IBM PC XT and 1MB on the AST with 384K acting as conventional memory and 64K as an extended print spooler. My Framework II setup configuration is Overlays = No, Buffer = 20, Extended Memory = On, Memory Driver = AST Rampage!, Max Size = 2048, and Memory Drive = C. Given the above I get 207,072 bytes free in Framework II after loading a shortened version of Sidekick. Shouldn't I have more memory available? I've read the messages here on how to set the extended memory and, unless I've missed something its setup as it should be, right?

In the July 1986 issue of ***TechNotes*** on page 55 it mentions that a person has both memory switches set to off. What does that mean?

I'll bet you get an error message during Framework II bootup that says something about not being able to use extended memory. You have set the size of extended memory to 2048, when you don't have that much available. Try changing it to 1536.

The two memory switches referred to are the memory configuration switches on the IBM PC XT motherboard. That user had used 256K chips to put all 640K on the motherboard and then set his configuration switches to tell the system how to deal with it.

.....

I am looking for a driver for the QuietWriter II that handles the text and all points addressable graphics. The QuietWriter driver in DL 3 doesn't seem to be compatible with Framework II. Do you happen to have one in your back pocket?

Configure your system as the IBM Graphics Printer since that is what the QuietWriter II emulates. Your printer will work fine with both text and graphics.

.....

MultiMate:

When using the Prop B Font on my HP LaserJet, I am having trouble with the right justification. We are printing resumes and the right margin will not justify even though it is set with tabs and carriage returns. Regular text is not a problem. Any suggestions?

MultiMate does not right-justify when there is a carriage return at the end of the line. The only thing that I can suggest is to external copy a soft column break symbol " <—" into your document. You can obtain a soft column symbol by repaginating in snake column mode.

.....

While using MultiMate Advantage 3.6, the HP LaserJet Plus we recently purchased prints about three pages a minute, but prints its normal eight or nine pages a minute with other programs. Anyone know what the problem might be? We are using the H Cartridge and doing nothing fancy.

Check these options on the "Print Parameter" screen.

Y Draft Print
N Micro
N Proportional Spacing
P Parallel - Com1 Redirected to LPT1

This should enable you to get maximum speed possible in print time with MultiMate on the HP LaserJet.

.....

Could you please supply some information on the easest ways to do envelopes with MultiMate 3.31? I am a new purchaser and none of the materal supplied seems to answer this very important question for me.

*The simplest way to do an envelope with MultiMate is to edit the letter with the address on it. Press **Ctrl-End** which will bring you to the bottom of that page and then **F2** to create a second page. If you have more than one page in your letter go to the bottom of the last page with text on it and complete the above. Go to the address on top of your letter and press **F8** and highlight the address using the arrow keys. Once all of the address is highlighted, hit **F8** again. Press **F1** to go to the blank page you just created and hit **F8** again. The address should now appear on the new page. Now hold down the **Shift** key while pressing **F3** and hit **Ins**. You will see a prompt at the bottom of the screen asking you at what column and how many lines, just fill in the appropriate info and hit **F10** key. Your address will now appear at the correct place to be printed. Specify a top margin and the length of the envelope on the "Print Parameters" screen.*

If you have a HP LaserJet Printer, refer to the Technical Support Library in the Display area, for MultiMate, under Common Questions for the HP.

.....

MULTIMATE TECHNICAL SUPPORT

Submitted by Liz Burke

MultiMate Magic

How can I print line and box characters in landscape mode on my Hewlett-Packard LaserJet printer?

Use the V cartridge. It is the only cartridge that can print these characters in landscape mode.

Selective Merge Print

I have a two-page primary merge document. Page 1 contains Merge items, and page 2 has only text. How can I Merge Print just page 1 on my company's letterhead.

To accomplish this, print page 1 and page 2 separately. Follow the steps below:

1. From the MultiMate main menu select option 5, **Merge Print Utility**.
2. Type in the name of the **MERGE DOCUMENT** and the **MERGE DATA FILE**.
3. Press **F10** twice to get to the "Print Parameters For Document" screen.
4. Enter 1 for the options **Start Print At Page Number** and **Stop Print After Page Number**.
5. Then, press **F10** to begin the merge print.

With this formulation, only page 1 of the primary document will print. Print page 2 after the first page of each document is finished. To do this select option 3 (**Print Document Utility**) from the MultiMate main menu. At the "Print Parameters For Document" screen, set **Start Print At Page Number** and **Stop Print After Page Number** to 2.

Drive Letters

Why doesn't MultiMate Advantage recognize the drive letter that I have assigned to my subdirectory?

You probably need to modify the drive defaults on the "Drive Default Modification" screen. MultiMate Advantage recognizes drive letters A through P. After you assign a drive letter to a subdirectory, select option 2 (**Edit Drive Defaults**) from the "Other Utilities Menu." This takes you to the "Drive Default Modification" screen. For the option **Installed Drive Table**, enter Y beneath each drive letter you have assigned.

Changing Spacing

How can I change a single-spaced document to double-spaced?

Use MultiMate Advantage's global replace function to replace the single-spaced format line with a double-spaced format line. The steps to do this are outlined below:

1. Move the cursor to the top of the document.
2. Press **Shift-F6**.
3. Press 1 for **Global Replace**.
4. Press **Shift-F9** to edit the current format line.
5. The first number on the format line indicates the line-spacing. Press **Home** to move the cursor to the beginning of the format line and then change that number to 2.
6. Press **F10** to change the format lines throughout your document.

Converting an Information Handling File to an ASCII File

The Information Handling feature in MultiMate Advantage is designed to make it easy for you to create lists, such as names and addresses. The records in an Information Handling document can be converted to the ASCII format if, for example, you wish to send the data across a modem, use it for a Merge Print, or transfer it to another software program.

The process for converting an Information Handling document to ASCII is different from that for a regular MultiMate document. The conversion utility (Util) that comes with MultiMate Advantage does not convert Information Handling files to ASCII format. However, you can get the same result by doing a Merge Print to an ASCII file instead of the printer. The following instructions outline how to do this.

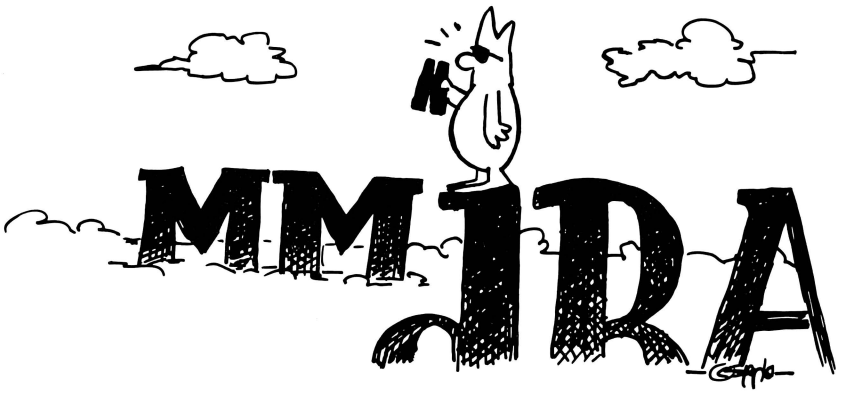
1. To begin, you must edit the Printer Action Table (PAT) for TTYCRLF.PAT. To do this, go to the drive or directory where Util resides and type,

   A > UTIL

2. At the first screen, press **F10** three times. This will take you to the "Edit An Old File" screen.
3. Highlight the option **TTYCRLF** and then press **F10**.
4. At the "Edit Printer Action Table" screen, highlight the option **Print Styles and Enhancements** and then press **F10**.
5. Since you do not want underlining control codes sent to the ASCII file, enter C5 for the **Enable** and **Disable** sequences of the **Underline** option.
6. Press **F10**. This takes you back to the "Edit Printer Action Table" screen.
7. Highlight the option **Save PAT File (with same name)**, and then press **F10**.
8. To return to DOS, press **Esc** three times.

If you use this TTYCRLF PAT for future printing, underlining will not be included in your documents.

You are now ready to print your Information Handling document to an ASCII file. Return to the MultiMate main menu, and follow these steps:



1. Select 5 **Merge Print Utility**.
2. From the "Merge Print a Document" screen, select the option **MERGE DOCUMENT** and enter the name of the Information Handling document.
3. Press **F10** twice and bypass the prompt "Record(s)" and move to the "Print Parameters for Document" screen. At this screen, set the defaults as follows:

1. Set the **Top Margin** and **Left Margin** to 0.
2. Enter Y for **Draft Print**.
3. Enter TTYCRLF for the option **Printer Action Table**.
4. Enter F for the option **P(arallel) / S(erial) / L(ist) / A(uxiliary) / F(ile)**.

4. Press **F10** twice to begin printing to an ASCII file.

Now the file is an ASCII text file. To see the contents of the file, issue the TYPE command from the DOS level.

Converting Document Content Architecture Files

The Document Content Architecture (DCA) file format is an IBM format that allows you to transfer documents between various office systems. There are two DCA formats: Revisable Form and Final Form. MultiMate's DCA conversion utility converts DCA Revisable Form files to MultiMate document files and vice versa. The DCA conversion allows you to combine MultiMate's power, versatility, and ease of use with the benefits of office-wide distributed word processing and data transfer. You can share documents among various office systems, including the IBM-PC, IBM Displaywriter, and systems that employ SNA (System Network Architecture) and IBM DISOSS (Distributed Office Support Systems).

Because the MultiMate DCA conversion utility is not a communication package, you must supply the hardware and software to transfer files between the IBM-PC and another computer. You must also provide any necessary conversions between DCA Final Form and Revisable Form formats.

There are several important issues to consider when you convert files to and from DCA format. Some MultiMate features are supported in DCA format and others are not. Similarly, some DCA format features are interpreted correctly when converted to a MultiMate document and others are not. This article outlines these differences and, if possible, offers work-arounds.

Requirements

To perform this conversion, you need to have the following:

1. MultiMate version 3.30 or higher.
2. At least 320K of RAM in your computer.
3. The MultiMate Advanced Utilities programs (UTIL.EXE and UTILMSG.TXT) and the file conversion files (DCA.OV1 and DCA.OV2) in the MultiMate directory.

Converting DCA Files to MultiMate Document Files

The DCA document merge function is not supported by MultiMate. In addition, there are differences in the way some of the DCA attributes are displayed, such as backspacing, extended characters, headers, and footers. These variations are outlined below.

Backspace Character: The DCA backspace character is converted to the MultiMate printer code for backspace, [00e6] 008 (**Alt-A** 008). When the MultiMate document is printed, backspacing occurs whenever the [00e6] 008 sequence is found.

Extended Characters: Foreign language characters that occur in both the DCA and IBM-PC character sets transfer correctly. However, characters that only occur in the DCA character set are converted to IBM-PC graphic characters, which do not accurately represent the original character. If the file is converted back to the DCA format, the characters are restored correctly.

Headers and Footers: Headers and footers from a DCA file are always placed on the first page(s) of the MultiMate document. After conversion, you may have to edit the MultiMate document to place these headers and footers on the appropriate lines and pages.

Merge: The merge-printing codes used in the DCA format are correctly converted to MultiMate merge codes. However, you may have to rearrange the codes and data before using the MultiMate merge-printing feature.

Converting a MultiMate Document to a DCA File

The following MultiMate functions are not supported in the DCA format: document merge, draft print, and enhanced print. In addition, when a MultiMate document is converted to a DCA file, differences occur with bold print, double-underline, extended characters, headers, footers, pitch changes, printer control codes, and shadow printing. These differences are outlined below.

Bold Print: Not all programs that accept DCA files use the same codes for bold printing. The MultiMate bold print codes are converted to the bold print codes used by DisplayWrite 3, not DisplayWrite 2.

Decimal Tabs: MultiMate decimal tabs are represented by standard tabs and spaces in the DCA file. For this reason, you may have to edit the DCA document to adjust the format line, tabs, and text.

Double Underline: Double-underlined MultiMate text is converted to single-underlined text in the DCA file.

Extended Characters: The MultiMate line and box drawing characters are not supported in the DCA format. However, foreign language characters that occur in both the DCA and IBM character sets transfer successfully.

File Extensions: Use .RF or .RFT as the filename extension, as required by the IBM system receiving the DCA file.

Date, Time, and Page: Automatic page numbering (# symbol) in a MultiMate header or footer will convert to automatic page numbering in the DCA format. However, MultiMate headers and footers can also contain the date, time, and the current/last page number, though they are not supported by the DCA format. If your headers or footers contain these features, remove them before converting the file.

Header and Footers: DCA supports up to four pairs of headers and footers: two even headers and footers and two odd headers and footers. Only the first four headers in a MultiMate document are accepted by the DCA file. In addition, the first four headers are paired with the first four footers.

In MultiMate, each header and footer can have its own format line. In the DCA format, a header/footer pair must share a common format line. If your MultiMate document has headers and footers that contain different format lines, edit the document so that the header/footer pairs share the same format line.

For more information regarding headers and footers, refer to the section above "Date, Time, and Page."

Merge: MultiMate merge-print codes become DCA merge-print codes, though you will have to edit the DCA file to reorder the codes and data for use with the DCA merge print function.

Print Change: MultiMate supports pitch change codes anywhere within text. However, the DCA format only supports a pitch change at the beginning of a line. When converted to the DCA format, any MultiMate pitch change codes after the first character of a line are ignored.

Printer Control Codes: The MultiMate printer control codes for backspace (**Alt-A** 008) are correctly converted to the DCA backspace character. All other printer control codes are replaced by an underscore, which is the DCA substitute control symbol.

Shadow Print: The DCA format does not support shadow printing. MultiMate shadow print codes are converted to bold in the DCA file. Refer to the above section "Bold Print" for more information.

In this article we have discussed the requirements for converting files to and from the DCA format, as well as the kinds of attribute changes to expect. For more information about using the DCA file conversion utility, refer to the *MultiMate Advanced User's Guide*.

Converting a Lotus 1-2-3 worksheet to a MultiMate Document File

MultiMate's conversion utility (Util) has the capability of converting 1-2-3 worksheet files, both .WKS and WK1, to MultiMate document files. This is a one-way conversion. That is, it does not convert MultiMate document files to either of the 1-2-3 formats.

Before using the 1-2-3 conversion utility, you do not have to change any of the default settings. However, you may have to prepare the worksheet before converting it or edit the MultiMate document after the conversion. As in "Converting Document Content Architecture Files," this article tells you how to successfully convert 1-2-3 files to MultiMate document files.

Requirements

To convert a 1-2-3 worksheet to a MultiMate document file, you need the following:

1. MultiMate version 3.30 or higher.
2. The MultiMate Advanced Utilities programs (UTIL.EXE and UTILMSG.TXT), the 1-2-3 conversion files (123.OV1), and the MultiMate program files (MM.OV1 and MM.OV2) in the same directory.

File and Page Size

A MultiMate document has a maximum size of 128K. Files larger than that are truncated during conversion. If your worksheet is too large, divide it into two separate files before converting it to MultiMate.

A single page in a MultiMate document file can contain 150 lines or 6,144 characters. The MultiMate file conversion program automatically starts a new page when it reaches either of these limits.

For more information on MultiMate document limitations, refer to Appendix C, "MultiMate Document Parameters," in your *MultiMate Reference Manual.*

Line Length

The maximum line length in MultiMate is 156 characters. If your 1-2-3 worksheet is wider, you must either change the width to fit the 156-character limitation or divide the worksheet into two files and convert them separately.

1-2-3 Graphics

The MultiMate conversion utility does not convert 1-2-3 graphics files (.PIC). 1-2-3 graphics can, however, be printed in MultiMate documents through the MultiMate GraphLink software. Refer to the GraphLink documentation for instructions on how to do this.

Rounding Off Decimals

The MultiMate file conversion maintains the column widths used in your 1-2-3 worksheet. MultiMate always aligns numbers from your worksheet using decimal tabs. In addition, the conversion process rounds off numbers that are too large for the column width. To avoid this, before converting the 1-2-3 worksheet, make sure the column width is wide enough to accommodate the number, plus one extra space. This extra space is for the MultiMate decimal tab character.

Scientific Notation

Numbers represented in scientific notation, such as 987E + 6, are converted as a line of asterisks (*********). If your worksheet uses scientific notation, send the 1-2-3 worksheet to an ASCII (.PRN) file, and then import it into MultiMate using the ASCII conversion program.

For more information on converting 1-2-3 worksheets to MultiMate document files, refer to the *MultiMate Advanced User's Guide.*

LETTERS

Dear ***TechNotes***:

I know that I can type DBASE < program > to run a command file from the DOS prompt. I also know that in my CONFIG.DB file, I can put COMMAND = DO < program > and that will also run a command file when I run dBASE III PLUS. In either case, before my program executes, the license agreement appears and I am prompted to press the **Return** key to continue. However, after a certain amount of time and the **Return** key has not been pressed, the program then executes. Is there a way to bypass the license agreement without the time-out? A second question, is there a way that I can simulate this time-out?

D.O.

Dear D.O.:

In answer to your first question, you cannot bypass the license agreement if you have either entered DBASE < program > or have COMMAND = DO < program > in your CONFIG.DB file. As you have surmised, the license agreement times out for a few seconds and then dBASE III PLUS executes your program.

In answer to your second question, you can use a counter in combination with a DO WHILE loop to create a time-out procedure. The following program demonstrates how to create a simple timing loop,

```
x = 0
DO WHILE x < 100
   x = x + 1
ENDDO
```

This basic concept can be enhanced using the INKEY() function to allow you to create a procedure that pauses for a given period of time or until a key is pressed. INKEY() is used to detect a keypress. If you press a key, INKEY() returns the ASCII value of the key, otherwise it returns 0. For example, the following is a key test program where the loop terminates only when a key is pressed.

```
i = 0
DO WHILE i = 0
   i = INKEY()
ENDDO
```

So, by incorporating the key test into the time-out procedure discussed you get a program that simulates the dBASE III PLUS license agreement time-out.

```
i = 0
x = 0
@ 10,10 SAY "Press any key to continue."
   DO WHILE i = 0 .AND. x < 100 i = INKEY()
   x = x + 1
ENDDO
@ 10,0
```

Here, the DO WHILE loop executes 100 times or if a key is pressed the loop terminates and the message is erased. You can of course, check for the key press of a specific key. As example, to check for the **Return** key, change the DO WHILE loop above to the following,

```
DO WHILE i < > 13 .AND. x < 100
```

SET FILTER

Dear TechNotes:

I notice that whenever I SET FILTER TO a condition and GO TOP with large database files, it takes a long time for the GO TOP to execute. What is going on?

J.N.

Dear J.N.:

SET FILTER TO <condition> shows only the records in the current database file that meet the condition you specify. When you issue GO TOP, dBASE III PLUS executes a LOCATE FOR <condition> internally. That is, it moves to the top of the database file and then moves sequentially through the database file searching for the first record that satisfies the condition. If you have a large database file where the first matching record is not the first record, then GO TOP takes as much time as it takes to process all the records up to the match. If there are no matching records in the database file at all, then GO TOP will take as much time as it takes to scan the entire database file. As you can see, SET FILTER TO may not be the most efficient method of processing your database file. However, there are some other methods.

Suppose you have a database file from which you are going to print labels. You have already INDEXed it on the zip code field to get better postal rates, and now you want to print all the labels for zip codes beginning with "95." For speedy execution of the LABEL FORM, you can move quickly to the first instance of "95" and then use the WHILE condition to print labels until there are no more zip codes beginning with "95." The following demonstrates how to do this,

```
SEEK "95"
LABEL FORM <your label file> WHILE <zip code> = "95"
```

To narrow the scope of the labels you are printing further, you can add a FOR clause to the LABEL FORM command line. For example, within the set of zip codes beginning with "95", you want to print only those records with the last name of "Smith." The following command lines do this.

```
SEEK "95"
LABEL FORM <your label file>;
        WHILE <zip code> = "95";
        FOR <last name> = "Smith"
```

As you see from this simple example, you can accomplish the equivalent actions to a SET FILTER by using a index file in combination with a WHILE condition. This is just one way you can circumvent the slowness of a FILTER.

CORRIGENDA

Two errors were found in the article about DIF files called "Exporting to a VisiCalc DIF File" in the September 1986 issue of ***TechNotes***. At the top of page 40,

TABLE 0,1' should be

TABLE
0,1

and

DATA 0,0 should be

DATA
0,0

READER SUBMISSION FORM

If you have a programming tip, a handy formula, or a short utility, and you are interested in sharing it with other Ashton-Tate or MultiMate enthusiasts in a future issue of ***TechNotes***, please read the provisions below. After reading these guidelines, fill out this form and send it along with your submission(s) to the following address. Submissions must be made on disk (IBM or compatible format) in standard ASCII format or in Framework II (.FW2) format.

***TechNotes* Submissions**
Software Support Center
Ashton-Tate
20101 Hamilton Ave.
Torrance, CA 90502-1319

NAME _____ DATE _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

PHONE (During normal business hours) (_____) _____ - _____

_____

(SIGN HERE TO ACKNOWLEDGE YOUR UNDERSTANDING OF THE PROVISIONS BELOW)

To establish a clear understanding of the rights and usage of submissions, you are asked to first represent and warrant that:

1. You are the author of the material: that it is original and has not been heretofore published; that it does not infringe upon any copyright, proprietary right or any other right whatsoever; that it contains no matter that is contrary to law; and that you are the sole owner of the rights herein conveyed to us, and that such rights are free and clear.
2. We shall have the right to use your name, biography or other information pertinent to your submission.
3. We shall have the right, at our discretion, to edit the material to suit the requirements of our publication.
4. Suggestions, comments, and programming materials submitted to ***TechNotes*** are not for reply. Materials submitted for publication are considered contributions to the "pool" of information found in ***TechNotes*** for the advantage of our readers. Although, we do not pay or in any other way compensate you for your contribution, your name shall appear as author or contributor.

Thank you for reading ***TechNotes***. If you do not have a submission for us, write us anyway and let us know what you think about our publication. We certainly appreciate any feedback you may have.

SUBSCRIBE TODAY TO *TechNotes!*

Don't miss a single issue of Ashton-Tate's hottest source of technical know-how! Each monthly issue of ***TechNotes*** provides you with the most current, accurate technical information available from the Software Support Center on all Ashton-Tate and MultiMate products.

Call our toll free number to subscribe:

(800) 331-4164 (In California)
(800) 828-2514 (Residents of other states)
(619) 487-3212 (In Canada)

If you prefer, fill out the form below and return it with payment to:

***TechNotes* Subscription Service**
P.O. Box 3729
Escondido, CA 92025-0929

— — — — — — — — — — **(CUT HERE)** — — — — — — — — — —

WE ACCEPT CREDIT ORDERS! Call our toll free number for details. Make checks payable to ASHTON-TATE.

Amount Enclosed:

__ $50.00
__ $65.00 (Extra Postage Canada and Mexico)
__ $75.00 (Extra Postage other foreign countries)
__ Bill Me

Form of Payment: (check one)

__ Check (**U.S. dollars drawn on U.S. bank**)
__ VISA Number _____
__ MasterCard Number _____
Expiration Date _____
Signature _____

Check One: __ New Subscription __ Renewal

Name _____

Company _____

Address _____

City _____ **State/Province** _____

Zip/Postal Code _____ **Country** _____

Which product(s) do you use? __ **dBASE II** (D2)
__ **dBASE III** (D3)
__ **Framework** (FW)
__ **MultiMate** (MM)

Please allow 30 to 60 days for delivery of your first issue.

1000

TechNotes
BACK ISSUES ORDER FORM

Maybe your subscription to ***TechNotes*** just began and you want to have every issue. Maybe you don't subscribe, but want to receive last month's issue. Maybe you want to refer to an article in a past issue to help you design that special application.

Whatever the reason, order Back Issues! Add to your library of dBASE II, dBASE III PLUS , and Framework II technical reference material. To order your Back Issue(s), fill out this form, enclose your payment, and return to:

***TechNotes* Back Issues**
Ashton-Tate Software Support Center
Attn: Kathy Smith
20101 Hamilton Ave.
Torrance, CA 90502-1319

Name _____

Company _____

Address _____

City _____ State/Province _____ Zip Code _____

Country _____ Phone Number (_____) _____-

All 1985 Back Issues are $10.00 each, when you order them separately. If you purchase all available 1985 issues as a set, the price is $50.00. All 1986 Back Issues are $7.00 each. All prices include postage and tax.

The following issues are currently available. (Those crossed out are out of print.) Check those you wish to order:

( ) FEBRUARY 1985
~~( ) MARCH 1985~~
~~( ) APRIL 1985~~
( ) MAY 1985
( ) JUNE 1985
( ) JULY 1985
~~( ) AUGUST 1985~~
~~( ) SEPTEMBER 1985~~
( ) OCTOBER 1985
( ) NOVEMBER 1985
( ) DECEMBER 1985 (includes 1985 Annual Disk and 1985 Cumulative Index)
( ) JANUARY 1986
( ) FEBRUARY 1986
( ) MARCH 1986
~~( ) APRIL 1986~~
( ) MAY 1986
( ) JUNE 1986
( ) JULY 1986
( ) AUGUST 1986
( ) SEPTEMBER 1986
( ) OCTOBER 1986

$_____ TOTAL AMOUNT ENCLOSED

( ) VISA ( ) MasterCard # _____

Expiration Date _____ Signature _____

ALL ORDERS MUST BE PREPAID. MAKE CHECKS PAYABLE TO ASHTON-TATE (IN U.S. DOLLARS DRAWN ON U.S. BANK). PLEASE ALLOW SIX (6) WEEKS FOR DELIVERY OF YOUR ORDER.

NOTE: Information contained in ***TechNotes*** is current as of the issue in which it appears. Therefore, some information found in the Back Issues may no longer be valid (such as Update Policies, etc.).

TechNotes
P.O. Box 3729
Escondido, CA 92025-0929

ADDRESS CORRECTION REQUESTED

BULK RATE
U.S. POSTAGE
PAID
PERMIT NO. 27
CULVER CITY, CA